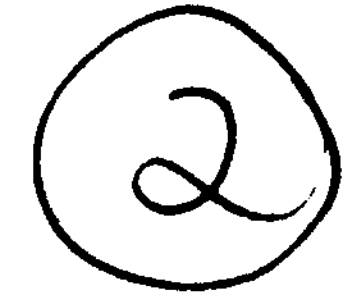

FINAL TECHNICAL REPORT

EXPENDABLE AIR VEHICLES/ HIGH ALTITUDE BALLOON TECHNOLOGY

2 August 1991

CHR/91-2750

Sponsored By:
Defense Advanced Research Projects Agency (DoD)
Defense Small Business Innovation Research Program
ARPA Order No. 6685

Issued By U.S. Army Missile Command
Under Contract Number:
DAAH01-90-C-0234

Prepared By:
Robert L. Hawkins, Principal Investigator
Coleman Research Corporation
6820 Moquin Drive
Huntsville, AL 35806

Contract Effective: 31 January 1990
Contract Expires: 2 August 1991

Coleman Research Corporation

HEADQUARTERS

5950 Lakehurst Drive
Orlando, FL 32819
(407) 352-3700
FAX: (407) 345-8616

ORLANDO DIVISION EAST

3045 Technology Parkway
Orlando, FL 32826-3299
(407) 249-7717
FAX: (407) 381-3980

HUNTSVILLE DIVISION

6820 Moquin Drive
Huntsville, AL 35806
(205) 922-6000
FAX: (205) 922-6053

LAUREL DIVISION

14502 Greenview Drive, Suite 206
Laurel, Maryland 20708
(301) 470-3839
FAX: (301) 776-5461

SOUTHWEST DIVISION

3 Butterfield Trail Blvd,
Suite 119
El Paso, TX 79906
(915) 772-4444
FAX: (915) 779-8754

WASHINGTON DIVISION

9302 Lee Highway,
Suite 800
Fairfax, VA 22031
(703) 934-7810
FAX: (703) 934-7800

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION	1b. RESTRICTIVE MARKINGS
Unclassified	N/A

2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT
N/A	Unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
N/A	

4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
CHR/91-2750	N/A

6a. NAME OF PERFORMING ORGANIZATION	6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION
Coleman Research Corp.	N/A	DARPA U.S. Army Missile Command

6c. ADDRESS (City, State, and ZIP Code)	7b. ADDRESS (City, State, and ZIP Code)
6820 Moquin Drive Huntsville, AL 35806	AMSMI-RD-DP-TT Redstone Arsenal, AL 35898-5244

8a. NAME OF FUNDING/SPONSORING ORGANIZATION	8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
DARPA		Contract Number DAAH01-90-C-0234

8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS			
1400 Wilson Blvd. Arlington, VA 22209-2308	PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO	WORK UNIT ACCESSION NO.
	N/A	N/A	N/A	N/A

11. TITLE (Include Security Classification)

Final Technical Report

12. PERSONAL AUTHOR(S)

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final Technical Report	FROM 1/31/90 TO 8/2/91	1991 August 2	

16. SUPPLEMENTARY NOTATION

17.	COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	N/A

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The development of a digital computer simulation for high-altitude, scientific balloon drift pattern prediction is described. Provision is made for the use of either wind forecast data or a worldwide, empirical atmospheric data base. The program operates on a Macintosh computer and produces on-screen drift patterns which may be imported into commercially available graphics programs for annotation and/or printing. The report includes a listing of the FORTRAN source code for the program.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS	

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code)	22c. OFFICE SYMBOL
Robert L. Hawkins, Principal Investigator	(205) 922-6000	

DD Form 1473, JUN 86

Previous editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

BALLOON DRIFT PATTERN SIMULATION (BDPS)
FINAL TECHNICAL REPORT

2 August 1991

CHR/91-2750

Reporting Period:

31 January 1990 through 2 August 1991

In Accordance With Contract Number

DAAH01-90-C-0234 under CDRL A004

Prepared for:

U.S. Army Missile Command

DARPA Project Office

Redstone Arsenal, AL 35889-5280

Prepared by:

Coleman Research Corporation

6820 Moquin Drive

Huntsville, AL 35806

THIS PAGE INTENTIONALLY LEFT BLANK

PREFACE

The work described in this Phase II SBIR Final Technical Report is the implementation of a capability which Coleman Research Corporation demonstrated during a Phase I SBIR (contract number DAAH01-90-C-0234). Both contracts were performed under the administrative overview of the USAMICOM DARPA Project Office at Redstone Arsenal, Alabama. Technical guidance was given by DARPA from their Arlington, Virginia headquarters.

Accession For	
NTIS	
DTIC	
Unannounced	
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	

ACKNOWLEDGMENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

FIGURE	TITLE	PAGE

THIS PAGE INTENTIONALLY LEFT BLANK

1.0 **INTRODUCTION**

Under the sponsorship of the Defense Advanced Research Projects Agency (DARPA), Coleman Research Corporation (CRC) has developed a Balloon Drift Pattern Simulation (BDPS). CRC developed this simulation software for digital computers as a product of a Phase II Small Business Innovative Research (SBIR) project. This report presents the approach which CRC took to develop this product.

DARPA is interested in exploiting high-altitude, expendable balloon-borne communication and surveillance capabilities for military applications. Balloons offer several unique capabilities when high-altitude balloon technologies are integrated with payloads employing the power-efficient, light-weight electronic technologies available today. In order to establish communication and surveillance systems design requirements and deployment schedules, DARPA must analyze upper-atmosphere drift pattern simulation results for a variety of balloon payload concepts.

The Phase II development of BDPS was the logical activity to follow CRC's Phase I SBIR program [1] in which we demonstrated the technical feasibility of predicting high-altitude balloon drift patterns using a digital computer simulation. CRC realized the Phase I goal by completing each of four contract objectives: performing a literature survey, developing and integrating atmosphere and balloon models into a Balloon Drift Pattern Simulation, developing drift pattern output formats, and exercising the BDPS to produce and analyze balloon drift patterns. Drift patterns were produced for two balloon configurations which were specified by DARPA. The first mission employed a zero-pressure balloon which floated at an altitude of 70,000 feet over 24 hours. The second mission used a super-pressure balloon which floated at 120,000 feet for a one-year period. The models were executed and display output produced on VAX computers.

The project objective of the Phase II contract was to develop a Macintosh-based BDPS for DARPA. The accomplishment of this overall objective depended upon the completion of specific technical objectives, which are discussed in the following sections of this report: (1) Develop a suitable atmosphere model; (2) Streamline and modify the trajectory model; (3) Install BDPS on the Macintosh; (4) Develop Macintosh graphics output; and (5) Develop BDPS documentation.

2.0 BDPS ATMOSPHERE MODEL

The first technical objective to be met was the development of a suitable atmosphere model for use within BDPS. Since BDPS was intended to be used as a deployment tool, the atmosphere model was of critical importance because the wind would have the most significant impact on balloon motion.

CRC decided to design the BDPS tool to be able to use either of two atmosphere mc ·ls options. The first option uses wind tables which the user must provide. The second option makes use of an empirical climate model for which we have included climate data files with the delivered software.

2.1 WIND TABLE OPTION

This option was provided to give the BDPS user a mechanism by which wind data from a variety of sources may be incorporated to support drift pattern prediction. The option requires the user to supply two files in a format which is specified in the User Manual appendix. These two files contain, respectively, east wind velocity and north wind velocity. The velocity parameters in each file are organized as functions of latitude, longitude, and altitude.

This option has been specifically designed to support retrieval of forecast data generated by the Navy's NOGAPS capability at the Fleet Numerical Oceanography Center in Monterey, California. The NOGAPS data is made available through distribution software called NODDS which operates on an IBM-PC-compatible computer. CRC wrote and tested PC-based routines to arrange downloaded NOGAPS data into the east and north wind tables previously described. We then transferred the tables to the Macintosh using Apple File Exchange, a conversion utility which is routinely supplied with Macintosh system software. The source code for the PC-based routines has been included with the BDPS source code in the Analyst Manual appendix.

Though the wind table option was designed for use with NOGAPS data, any other data which is similarly formatted may be incorporated by using this atmosphere model option. For example, if a user had access to forecast wind data from another source or to archived wind data, the user could arrange the data into the two wind files and then use that data with BDPS. Furthermore, a user with access to a climate model could format the model's output appropriately to use that climate model data with BDPS.

2.2 CLIMATE DATA OPTION

This atmosphere model option was provided to give the BDPS user access to an empirical climate model. The model is based on NASA's Global Reference Atmosphere Model (GRAM), which is an empirical FORTRAN computer simulation of the earth's atmosphere. It was developed by the Georgia Institute of Technology under contract to NASA's Marshall Space Flight Center. Justus [2] summarizes the capabilities and operations of GRAM. The latest version of GRAM which was available during our development was the 1988 version (GRAM-88) [3].

The GRAM feature which is most significant to BDPS is that GRAM provides a worldwide, 12-month database of atmospheric properties including wind speed and wind azimuth. The wind data which GRAM provides may therefore be considered to be 4-dimensional because it is a function of latitude, longitude, altitude, and time of year. The empirical data is segmented into twelve files (one for each month) and is stored in a binary form for fastest access during execution of the model. The size of each of these files is approximately 3 megabytes.

CRC had to modify GRAM to make it suitable for use as an atmospheric model in BDPS. The baseline version of GRAM available from NASA operates in "batch" mode, whereby position input is supplied and GRAM then produces an output file containing an atmospheric profile for that position. CRC learned that Mr. Larry Schilling, working on the National AeroSpace Plane (NASP) program at NASA/Ames-Dryden, had modified GRAM-88 to work in an "interactive" mode with a NASP trajectory simulation program. Mr. Schilling was able to obtain improved performance by replacing the worldwide, empirical, monthly data files with smaller empirical data sets which only contained data for the contiguous United States (CONUS). Though GRAM-88 was designed to access the large empirical files from disk, Mr. Schilling loaded his CONUS files into memory, thereby optimizing atmosphere data access during the execution of his NASP simulation. Mr. Schilling modified the GRAM-88 structure and integrated it within his simulation to provide an interactive capability in which GRAM-88 was called at each integration step to provide new atmosphere data to be included in the equations of motion for the NASP. Because of the performance improvements and the throughput of the NASP simulation host computer, Mr. Schilling named his version "RTGRAM," which is an abbreviation for "Real-Time GRAM."

3

As shown in Figure 2.2-1, the climate model which CRC developed for BDPS used significant portions from both of the versions of GRAM which we obtained from NASA. From the perspective of BDPS requirements, each version had advantages and disadvantages. GRAM-88 had the advantage of worldwide data and the disadvantage of the batch mode of operation. RTGRAM had the advantage of interactive operation with a flight simulation and the disadvantage of empirical data limited to CONUS. Therefore, CRC's climate model for BDPS was built around the interactive structure of RTGRAM while using the worldwide empirical data files from GRAM-88. Though the CONUS data access techniques of RTGRAM would have provided a faster-running climate model, CRC could not reasonably expect to be able to load the entire worldwide file into memory on an average Macintosh computer. The worldwide file was required (and the CONUS file was inadequate) because DARPA personnel had advised us that, for political considerations, the balloon systems for which BDPS was intended would likely be tested in the southern hemisphere.

GRAM88
● WORLD-WIDE
● BATCH

(NASA / MSFC)

RTGRAM
● CONUS
● INTERACTIVE

(NASA / AMES)

BDPS CLIMATE MODEl
● WORLD-WIDE
● INTERACTIVE

Figure 2.2-1. BDPS Climate Model Heritage

The merging of the two versions of GRAM was a difficult task because of the poor quality of the GRAM source code. The resulting source code is included in the Analyst Manual appendix. CRC had hoped to acquire GRAM-90, but its

release was delayed by NASA until one month from the end of our Phase II contract. GRAM-90 offers improvements in its source code quality and in its southern hemisphere data. The features of GRAM-90 are described in publications by Justus [4] and [5].

3.0 **BDPS TRAJECTORY MODEL**

The second technical objective to be met was the development of a trajectory model for use in BDPS. The capability which was demonstrated in Phase I was inappropriate for Phase II for two reasons. First, the trajectory model was developed using a simulation framework which was proprietary to CRC. Since the Phase II deliverable product includes source code, CRC had to replace the proprietary portions of the trajectory model. Second, the Phase I approach modeled balloon dynamics to a level of detail which was inappropriate for a deployment tool on a Macintosh host. The Phase II deliverable product had to be streamlined in order to provide a useable tool on a platform with much less computational throughput than the platform used for the Phase I project.

The two major design constraints in the development of the trajectory model for BDPS are that (1) the model should include the detailed effects of buoyancy variations through the balloon's ascent and the effects of winds produced by the atmosphere model and (2) the level of modeling detail should be minimized to produce a tool that is useable on the Macintosh. The following paragraphs describe the approach CRC took in balancing the high fidelity and low detail requirements.

In the Phase I literature survey, CRC found that balloon designers already had detailed thermodynamic models [6], [7] of balloon ascent (vertical motion). However, we found no published information about balloon simulations which considered horizontal motion. Given this information and DARPA's stated need for a deployment tool, CRC decided to focus on wind-induced motion as the major consideration for BDPS.

As we were formulating ideas and approaches for a BDPS trajectory model, we noticed a consistent trend in the output from a number of different test runs produced with our Phase I tool on the VAX. Figures 3.0-1 through 3.0-4 show sample wind profiles produced under four separate sets of circumstances: Wallops Island, Virginia in January; Wallops Island in August; Vandenberg AFB, California in January, and Vandenberg AFB in August. We noted that the four wind magnitude profiles showed significant variations from each other. However, Figures 3.0-5 through 3.0-8 show that the vertical motion for a single

balloon configuration in each of those four circumstances was essentially constant. A comparison of the numbers in the various plots revealed a maximum difference on the order of 1%.

Figure 3.0-1. Wind Profile for Wallops Island in January

Figure 3.0-2. Wind Profile for Wallops Island in August

Figure 3.0-3. Wind Profile for Vandenberg AFB in January

Figure 3.0-4. Wind Profile for Vandenberg AFB in August

8

Figure 3.0-5. Ascent Profile for Wallops Island In January

Figure 3.0-6. Ascent Profile for Wallops Island In August

Figure 3.0-7. Ascent Profile for Vandenberg AFB in January

Figure 3.0-8. Ascent Profile for Vandenberg AFB in August

From the analysis of the data depicted in Figures 3.0-1 through 3.0-8, we concluded that the vertical motion of a balloon configuration is essentially independent of the horizontal wind profile with which the balloon motion is simulated. Detailed thermodynamic models [6], [7] may be used to generate the vertical motion profile for a particular balloon configuration. Furthermore, Fichtl [8] had stated that a balloon's horizontal wind-relative velocity (i.e., the difference in the balloon velocity and the wind velocity) is zero after a short time

10

for settling of transients. For the design of the BDPS trajectory model, we decided that the balloon's three-component velocity vector could be formed by the combination of the east and north wind velocity vectors taken from the atmosphere model (either the wind table or climate option) and the vertical velocity which may be generated offline from a more detailed, balloon design tool. By decoupling the vertical and horizontal velocity of the balloon motion, CRC has produced a simplified trajectory model which simultaneously satisfies the two design constraints stated above.

CRC tested the results of the simplified trajectory model introduced herein against the Phase I capability. The results showed acceptable agreement between the two methods. The largest observed percent difference between the two methods was less than 1%.

4.0 BDPS MACINTOSH INSTALLATION

The third technical objective to be met was the installation of the BDPS on the Macintosh. This involved four steps which are listed below and described in the following paragraphs: (1) Transfer source code, climate data files; (2) Compile and link FORTRAN code on Macintosh; (3) Duplicate sample test cases; and, (4) Add graphical user interface.

4.1 TRANSFER SOURCE AND DATA

CRC used standard file transfer tools (Kermit, Xmodem, etc.) to transfer the FORTRAN source code for BDPS from the VAX to the Macintosh. The transfer of the twelve monthly climate files, however, was more involved. For efficiency reasons already discussed, the data are stored in binary form for use by GRAM. Each of the files had to be converted to an ASCII representation on the VAX, transferred in ASCII form to the Macintosh, and then restored to binary form on the Macintosh. The conversion on the VAX and the subsequent restoration on the Macintosh were both accomplished by using very simple FORTRAN programs which were compiled and linked on the respective machines. While the binary form of each climate file occupied ~3 megabytes, the ASCII form required approximately 10 megabytes per file.

4.2 COMPILE AND LINK ON MACINTOSH

The VAX environment for BDPS development used the VMS 5.4 operating system, the VAX FORTRAN compiler, and the VAX Symbolic Debugger on a VAX 3900 machine with DEC VT-240 terminals. CRC used two Macintosh development environments for BDPS. The first was a Macintosh IIfx running System 6.0.7. On the IIfx, CRC used Apple's Macintosh Programmer's Workshop (MPW) version 3.1, the Language Systems FORTRAN compiler version 2.1, and Apple's Symbolic Application Debugging Environment (SADE) version 1.1. The second system was a Macintosh SE/30 running System 7.0, MPW 3.2, SADE 1.3, and the same FORTRAN compiler.

CRC encountered numerous problems in the creation of a BDPS version on the Macintosh. We had already known that the VAX and the Macintosh were internally different with respect to numeric data representation and alignment. However, the poor quality of the GRAM source code proved to be a major obstacle in producing a working BDPS version on the Macintosh. In particular, the GRAM source code flagrantly violates professional FORTRAN programming standards in its haphazard use of common blocks. The GRAM code had the

distinct appearance that the various subroutines had been written by different people who engaged in little, if any, coordination while developing their separate pieces.

The SADE debugger was of some help in tracing problems on the Macintosh. However, our use of SADE and the VAX debugger in simultaneous debugging sessions consistently convinced us that the SADE tool, while helpful, was not as robust as the VAX debugger. The net result of this difference is that we spent more time debugging on the Macintosh than we would have otherwise. We are encouraged, however, by some substantial improvements in the functionality of SADE 1.3 as compared to that of SADE 1.1.

4.3 DUPLICATE VAX RESULTS ON MACINTOSH

Once we had a fully functional version of BDPS on the Macintosh, we conducted several tests using the same data files on each of the two platforms. The results of the tests matched each other except for discrepancies in the lower-order digits of parameters. We attribute this difference in precision to the different mechanisms which the two platforms use in floating-point calculations. The VAX uses 32-bit representations for single precision throughout its operations. However, the Macintosh converts the 32-bit floating-point numbers to and from 80-bit numbers for operations which involve the 68882 floating-point co-processor in the IIfx and the SE/30.

4.4 ADD GRAPHICAL USER INTERFACE

In the statement of work for the Phase II contract, CRC stated that the interface for the Macintosh-based BDPS tool would be minimal in that it would not be designed to have the appearance and operation of well-known, commercially available Macintosh applications. However, CRC acquired a copy of Prototyper 3.0, which is a Macintosh user-interface development tool. We used this tool to create a Macintosh dialog box for configuring a BDPS run. The dialog box is shown in Figure 4.4-1. The most significant feature of the tool is that it can then generate compilable source code for the dialog items and their operation. CRC used the tool to generate Pascal source code and then converted the code to FORTRAN for consistency with the majority of the BDPS code. The operation of the user interface will be described in the User Manual appendix.

Edit

Mission Label: Wallops Island flight - Z configuration (this text may be used to identify the mission represented by this data)

Launch Position

Latitude: 37.9   deg

Longitude: 75.5   ● deg West   ○ deg East

Initial Altitude: 1.0   ● m   ○ km

Flight Duration: 24.0   ○ sec   ○ min   ● hr

Wind Model: January Climate ▼

Input Ascent Profile: Ascent Profile ▼

Run

Save

Map

Close

Figure 4.4-1.   Sample Dialog Box for BDPS Run Configuration

5.0 **BDPS MACINTOSH DISPLAY**

The fourth technical objective to be met was the development of a capability to display the drift pattern output on the Macintosh screen. During Phase I, CRC had used a VAX-based graphics package which generated pen-plotter output. For Phase II, the goal was to provide a capability on the Macintosh that would be an integrated part of the Macintosh-based BDPS and would produce drift pattern graphics that could be imported into commercially available Macintosh graphics packages.

CRC chose to save the on-screen drift patterns in the PICT file format which is supported by the major graphics packages. With this approach, the BDPS user can predict a drift pattern, display it on the Macintosh screen, and save it in a graphics file. The user can then use a commercial graphics package to open the saved graphic, annotate it as desired, and use the commercial package's printing capability to generate a hardcopy of the drift pattern.

The drift pattern latitude-longitude format was developed in Phase I and retained for use in Phase II. The display includes a world-map background which shows the physical outlines of the world's major land masses along with several islands. A sample display is shown in Figure 5.0-1.

File   Edit   Map   [?]

Map Window

Figure 5.0-1.   Sample Macintosh Drift Pattern Display

The on-screen display was developed by making FORTRAN calls to the Macintosh utility library which is present in the ROM and system software of every Macintosh.   These routines are extensively documented in <u>Inside Macintosh</u>, volumes I through V [9].   The source code for these display routines is included in the Analyst Manual appendix.

6.0 **BDPS DOCUMENTATION**

The fifth and final technical objective to be met was the development of documentation for BDPS. CRC chose to present the BDPS information from two perspectives: an Analyst Manual and a User Manual. The Analyst Manual presents the engineering design approach which CRC took in developing BDPS. The Analyst Manual presents this information by listing the source code which was used to generate the BDPS application on the Macintosh. The User Manual describes the operation of the drift pattern data generation and display modes of BDPS. Both of these manuals are included as appendices to this final report.

7.0 CONCLUSIONS AND RECOMMENDATIONS

CRC has successfully developed a Balloon Drift Pattern Simulation (BDPS) tool for use on the Macintosh computer. The BDPS tool is operated through a graphical interface similar to commercial Macintosh software. The drift pattern data is generated by using one of two options for supplying wind data: a climate model based on NASA'a Global Reference Atmosphere Model, or a table-driven wind model through which the BDPS user can provide wind data from another source. The resulting balloon drift pattern may be displayed on-screen and then imported into one of several commercial Macintosh graphics programs for annotation and/or printing. All source code for the BDPS tool has been made available through an appendix to this report.

The sole recommendation which CRC has to make regarding BDPS is that the performance be improved to make the tool more convenient to use. CRC has tested BDPS on a Macintosh SE/30 and on a Macintosh IIfx. On both systems, the process of generating drift pattern data (prior to the display of the drift pattern) requires several minutes. Our testing indicates that the GRAM model's heavy dependence on external files is the primary performance bottleneck. This problem could be mitigated by either of two approaches or by the combination of both approaches. The first approach would be to restructure the GRAM model to produce ·more efficient runtime operation. This process should begin with GRAM-90, NASA's newest version, which was not publically released until the end of CRC's period of performance in the development of BDPS. The second approach would be to rehost BDPS on a more powerful computer such as an engineering workstation. Running BDPS on a more powerful computer would improve performance regardless of whether GRAM had been restructured.

8.0 **REFERENCES**

[1] Coleman Research Corporation, "Expendable Air Vehicles / High Altitude Balloon Technology Final Technical Report," 28 February 1989, CHR/89-1909.

[2] Justus, C.G., Fletcher, G.R., Gramling, F.E., and Pace, W.B., "The NASA/MSFC Global Reference Atmosphere Model - MOD 3 (with Spherical Harmonic Wind Model)," NASA CR-3256, 1980.

[3] Justus, C.G., Alyea, F.N., Cunnold, D.M., Blocker, R.S., and Johnson, D.L., "GRAM-88: Improvements in the Perturbation Simulations of the Global Reference Atmospheric Model," NASA Marshall Space Flight Center Memorandum ES44-11-9-88, November 1988.

[4] Justus, C., and Johnson, D., "Extensive Middle Atmosphere (20-120 km) Modification in the Global Reference Atmospheric Model (GRAM-90)," 28th Aerospace Sciences Meeting, AIAA 90-0477, January 1990.

[5] Justus, C.G., Alyea, F.N., Cunnold, D.M., Jeffries, W.R. III, and Johnson, D.L., "The NASA/MSFC Global Reference Atmospheric Model – 1990 Version (GRAM-90)", NASA TM 4268, April 1991.

[6] Carlson, L.A., and Horn, W.J., "A Unified Thermal and Vertical Trajectory Model for the Prediction of High Altitude Balloon Performance," Texas Engineering Experiment Station Report TAMRF-4217-81-01, June 1981.

[7] Horn, W.J., and Carlson, L.A., "THERMTRAJ: A FORTRAN Program to Compute the Trajectory and Gas and Film Temperatures of Zero Pressure Balloons," Texas Engineering Experiment Station Report TAMRF-4217-81-02, June 1981.

[8] Fichtl, G.H., "Spherical Balloon Response to Three-Dimensional Time-Dependent Flows," NASA TN D-6829, July 1972.

[9] Apple Computer, Inc., Inside Macintosh, Volumes I-V, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1985.

THIS PAGE INTENTIONALLY LEFT BLANK

10.0 **BDPS ANALYST MANUAL**

The purpose of this appendix to the final report is to provide insight into Coleman Research Corporation's engineering approach and software implementation for the Balloon Drift Pattern Simulation (BDPS). Because the main body of the final report presents sufficient information about the engineering approach, this appendix presents the BDPS software details through the inclusion of the FORTRAN source code and associated utilities.

10.1 BDPS GENERATION SCRIPTS

As described in section 4.2, CRC used Apple's Macintosh Programmer's Workshop (MPW) and the Language Systems Corporation FORTRAN compiler. CRC automated the compilation and linking processes with scripts which are included h re.

10.1.1 **BDPS Compilation Script**

CRC used the following compilation script to in the generation of the BDPS object code modules. The script uses three directories: a source-code directory (in which FORTRAN filenames end with ".f", an include-file directory in which the filenames end with ".inc", and an object-code directory in which the compiler's output files are placed. Two GRAM-related files, "run.f" and "gen4d.f," are compiled with default 4-byte integer sizes. The other files are compiled with default 2-byte integer sizes because that size is more commonly used with Macintosh Toolbox arguments.

```
0001    #
0002    # bdpsCompile -    compile any source code targeted for bdps if either the
0003    #                  object code doesn't exist or the source code is newer
0004    #
0005    #   Targets every file whose name ends in .f
0006    #   in the source code directory
0007    #
0008
0009    SetDirectory Mars:darpa:MacBDPS:SourceFiles:
0010
0011    set FortranOptions          "-b -i Mars:darpa:MacBDPS:IncludeFiles: ∂
0012                                -i2 -mc68030 -mc68882 -nodyn -opt=1    ∂
0013                                -saveall -sane -sym -u"
0014    set FortranOptionsAlternate "-b -i Mars:darpa:MacBDPS:IncludeFiles: ∂
0015                                   -mc68030 -mc68882 -nodyn -opt=1     ∂
0016                                -saveall -sane -sym -u"
0017
0018    # compile run.f and gen4d.f with alternate compilation (for I*4)
0019
0020    if  `exists Mars:darpa:MacBDPS:ObjectFiles:run.f.o`
0021       if  Newer run.f Mars:darpa:MacBDPS:ObjectFiles:run.f.o
0022          fortran run.f {FortranOptionsAlternate} ∂
0023                      -o Mars:darpa:MacBDPS:ObjectFiles:run.f.o
0024       end if
0025    else if not `exists Mars:darpa:MacBDPS:ObjectFiles:run.f.o`
0026          fortran run.f {FortranOptionsAlternate} ∂
```

```
0027                           -o Mars:darpa:MacBDPS:ObjectFiles:run.f.o
0028      end if
0029
0030      if `exists Mars:darpa:MacBDPS:ObjectFiles:gen4d.f.o`
0031         if `Newer gen4d.f Mars:darpa:MacBDPS:ObjectFiles:gen4d.f.o`
0032            fortran gen4d.f {FortranOptionsAlternate} ∂
0033                           -o Mars:darpa:MacBDPS:ObjectFiles:gen4d.f.o
0034         end if
0035      else if not `exists Mars:darpa:MacBDPS:ObjectFiles:gen4d.f.o`
0036            fortran gen4d.f {FortranOptionsAlternate} ∂
0037                           -o Mars:darpa:MacBDPS:ObjectFiles:gen4d.f.o
0038      end if
0039
0040      # compile everything else (run.f and gen4d.f will be checked and passed)
0041
0042      for file in ≈.f
0043          set ObjectFile "Mars:darpa:MacBDPS:ObjectFiles:{file}.o"
0044          set SourceFile "Mars:darpa:MacBDPS:SourceFiles:{file}"
0045          if `exists {ObjectFile}`
0046              if `Newer {SourceFile} {ObjectFile}`
0047                  fortran {SourceFile} {FortranOptions} -o {ObjectFile}
0048              end if
0049          else if not `exists {ObjectFile}`
0050                  fortran {SourceFile} {FortranOptions} -o {ObjectFile}
0051          end if
0052      end
```

10.1.2 BDPS Linking Script

CRC used the following linking script to accomplish two objectives. First, the script compiles the BDPS resource file (if necessary). Then, the script links the previously compiled object code files with the required libraries to produce an executable application.

```
0001      # bdpsLink - link the object code in Mars:darpa:MacBDPS:ObjectFiles: to form
0002      #                   application bdps
0003      #
0004
0005      set RawRezFile      "Mars:darpa:MacBDPS:RezFiles:bdps.r"
0006      set CompiledRezFile "Mars:darpa:MacBDPS:bdps.rsrc"
0007
0008      if `Newer {RawRezFile} {CompiledRezFile}`
0009        SetDirectory Mars:darpa:MacBDPS:RezFiles
0010        rez -o Mars:darpa:MacBDPS:bdps.rsrc -c MDoF bdps.r ∂
0011             -i Mars:MPW:Interfaces:RIncludes:
0012      end if
0013
0014      SetDirectory Mars:darpa:MacBDPS:
0015
0016      duplicate -y bdps.rsrc bdps
0017
0018      Link -ac 4 -ad 4 -f -srt -sym on -w -o bdps          ∂
0019             Mars:darpa:MacBDPS:ObjectFiles:≈.f.o           ∂
0020             {Libraries}Runtime.o                           ∂
0021             {Libraries}Interface.o                         ∂
0022             {FLibraries}OutPWStubs.o                       ∂
0023             {FLibraries}FortranLib.o                        ∂
0024             {FLibraries}IntrinsicLibFPU.o                  ∂
0025             {FLibraries}FSANELibFPU.o
```

10.2 BDPS INCLUDE FILES

This section lists the contents of the "include" files which were made available to the FORTRAN compiler during compilation of the BDPS FORTRAN source code. Include files were generally used for the definition of common blocks and for the declaration of variable and static parameters which were referenced within multiple source code files. The filename for each include file is given in the commented first line of each file.

```
0001    c.....Alert.inc
0002
0003    c.....Alert declarations
0004
0005          integer*2              rAboutAlert
0006
0007    c.....Alert pre-settings
0008
0009          parameter             ( rAboutAlert          = 128 )


0001    c.....AppleMenu.inc
0002
0003    c.....Apple menu declarations
0004
0005          integer*2              nAppleItems
0006          integer*2              AppleItemAboutBDPS
0007          integer*2              AppleItemHelp
0008          integer*2              AppleMenuID
0009          record / MenuHandle / AppleMenuHndl
0010
0011    c.....Apple menu common block
0012
0013          common / AppleMenu / AppleItemAboutBDPS,        AppleItemHelp,
0014          .                    AppleMenuID,
0015          .                    AppleMenuHndl
0016
0017    c.....Apple menu pre-settings
0018
0019          parameter             ( nAppleItems          =   2 )
0020          parameter             ( AppleItemAboutBDPS   =   1 )
0021          parameter             ( AppleItemHelp        =   2 )
0022          parameter             ( AppleMenuID          = 128 )


0001    c.....CrvDat.inc
0002
0003    c.....Map curve parameters
0004
0005          parameter             (ncrvmx=4)
0006
0007          common / crvdat / ipvari,      ipvard,      ipvarg,      idrlin,
0008          .                 lintyp,      idrsym,      symtyp,      iplogx,
0009          .                 iplogy,      ipstep,      DshMsk,      PixCnt,
0010          .                 ighoff,
0011          .                 gradat
0012
0013          integer*2              ipvari(ncrvmx)
0014          integer*2              ipvard(ncrvmx)
0015          integer*2              ipvarg(ncrvmx)
```

```
0016            integer*2            idrlin(ncrvmx)
0017            integer*2            lintyp(ncrvmx)
0018            integer*2            idrsym(ncrvmx)
0019            integer*2            symtyp(ncrvmx)
0020            integer*2            iplogx(ncrvmx)
0021            integer*2            iplogy(ncrvmx)
0022            integer*2            ipstep(ncrvmx)
0023            integer*2            DshMsk(ncrvmx)
0024            integer*2            PixCnt(ncrvmx)
0025            integer*2            ighoff(ncrvmx)
0026
0027            real*4               gradat(ncrvmx)



0001    c.....DefLim.inc
0002
0003    c.....Map data common block
0004
0005            common / DefLim / LatMin,        LatMax,        LatDivMj,      LatDivMi,
0006            .                 LngMin,        LngMax,        LngDivMj,      LngDivMi
0007
0008            real*4            LatMin
0009            real*4            LatMax
0010            real*4            LatDivMj
0011            real*4            LatDivMi
0012            real*4            LngMin
0013            real*4            LngMax
0014            real*4            LngDivMj
0015            real*4            LngDivMi



0001    c.....EditMenu.inc
0002
0003    c.....Edit menu declarations
0004
0005            integer*2             nEditItems
0006            integer*2             EditItemUndo
0007            integer*2             EditItemCut
0008            integer*2             EditItemCopy
0009            integer*2             EditItemPaste
0010            integer*2             EditItemClear
0011            integer*2             EditMenuID
0012            record / MenuHandle / EditMenuHndl
0013
0014    c.....Edit menu common block
0015
0016            common / EditMenu /   EditItemUndo,          EditItemCut,
0017            .                     EditItemCopy,          EditItemPaste,
0018            .                     EditItemClear,
0019            .                     EditMenuID,
0020            .                     EditMenuHndl
0021
0022    c.....Edit menu pre-settings
0023
0024            parameter           ( nEditItems      =   5 )
0025            parameter           ( EditItemUndo    =   1 )
0026            parameter           ( EditItemCut     =   2 )
0027            parameter           ( EditItemCopy    =   3 )
0028            parameter           ( EditItemPaste   =   4 )
0029            parameter           ( EditItemClear   =   5 )
0030            parameter           ( EditMenuID      = 130 )
```

24

```
0001    c.....FileInfo.inc
0002
0003    c.....File information declarations
0004
0005        logical*1              iGotOldFile
0006        integer* 2             RefNum
0007
0008    c.....File information common block
0009
0010        common / FileInfo /
0011        &                      iGotOldFile, RefNum
```

```
0001    c.....FileMenu.inc
0002
0003    c.....File menu declarations
0004
0005        integer*2              nFileItems
0006        integer*2              FileItemNewMission
0007        integer*2              FileItemOpenMission
0008        integer*2              FileItemClose
0009        integer*2              FileItemSave
0010        integer*2              FileItemSaveAs
0011        integer*2              FileItemPageSetup
0012        integer*2              FileItemPrint
0013        integer*2              FileItemQuit
0014        integer*2              FileMenuID
0015        record / MenuHandle / FileMenuHndl
0016
0017    c.....File menu common block
0018
0019        common / FileMenu /   FileItemNewMission,     FileItemOpenMission,
0020        &                     FileItemClose,          FileItemSave,
0021        &                     FileItemSaveAs,         FileItemPageSetup,
0022        &                     FileItemPrint,          FileItemQuit,
0023        &                     FileMenuID,
0024        &                     FileMenuHndl
0025
0026    c.....File menu pre-settings
0027
0028        parameter             ( nFileItems          =   8 )
0029        parameter             ( FileItemNewMission  =   1 )
0030        parameter             ( FileItemOpenMission =   2 )
0031        parameter             ( FileItemClose       =   4 )
0032        parameter             ( FileItemSave        =   5 )
0033        parameter             ( FileItemSaveAs      =   6 )
0034        parameter             ( FileItemPageSetup   =   9 )
0035        parameter             ( FileItemPrint       =  10 )
0036        parameter             ( FileItemQuit        =  12 )
0037        parameter             ( FileMenuID          = 129 )
```

```
0001    c.....FntCom.inc
0002
0003    c.....font characteristics
0004
0005        common   FntCom                       FontData,    FntNam,        FntNum
0006
0007        record . FontInfo     FontData
0008        string*255            FntNam
0009        integer*2             FntNum
```

```
0001    c.....Globals.inc
0002
0003    c.....Globals declarations
0004
0005         integer*4              inFront
0006         integer*2              SleepValue
0007
0008         logical*1              gInBackground
0009         logical*1              gHasWaitNextEvent
0010         logical*1              doneFlag
0011
0012         record / TEHandle    / theInput
0013         record / SysEnvRec   / gMac
0014
0015    c.....Globals common block
0016
0017         common / Globals    /
0018         .                      SleepValue,
0019         .                      gInBackground,
0020         .                      gHasWaitNextEvent,
0021         .                      doneFlag,
0022         .                      theInput,
0023         .                      gMac
0024    c.....Globals pre-settings
0025
0026         parameter             ( inFront        = -1 )
0027         parameter             ( SleepValue     = 40 )
```

```
0001    c.....LatCom.inc
0002
0003    c      common block containing array of latitude values
0004
0005         common / LatCom /                Latitude
0006
0007         real                    Latitude(13120)
```

```
0001    c.....LngCom.inc
0002
0003    c      common block containing array of longitude values
0004
0005         common / LngCom /                Longitude
0006
0007         real                    Longitude(13120)
```

```
0001    c.....MapCom.inc
0002
0003    c.....Map data common block
0004
0005         parameter             (nptsmx=2048)
0006         parameter             (nvrsmx=  64)
0007
0008         common   Mapcom   MapWidth,   MapHeight,  MapHres,   MapVres,
0009         .                  DefWidth,   DefHeight,
0010         .                  TimeTics,   GridLines,  LimitType,
0011         .                  npts,       nvrs,       nMaps
0012    c    .                  ,header
0013
0014         real*4                MapWidth
0015         real*4                MapHeight
0016         real*4                MapHRes
```

26

```
0017          real*4          MapVRes
0018          real*4          DefWidth
0019          real*4          DefHeight
0020
0021          integer*2       TimeTics
0022          integer*2       GridLines
0023          integer*2       LimitType
0024
0025          integer*4       npts
0026          integer*4       nvrs
0027          integer*4       nMaps
0028
0029    c     string*255      header


0001    c.....MapLim.inc
0002
0003    c.....map limits and divisions
0004
0005          common / MapLim / xMapMn,    xMapMx,    xDivMj,    xDivMi,
0006          .                 yMapMn,    yMapMx,    yDivMj,    yDivMi,
0007          .                 tMapMn,    tMapMx,    tDivMj
0008
0009          real*4          xMapMn
0010          real*4          xMapMx
0011          real*4          xDivMj
0012          real*4          xDivMi
0013          real*4          yMapMn
0014          real*4          yMapMx
0015          real*4          yDivMj
0016          real*4          yDivMi
0017          real*4          tMapMn
0018          real*4          tMapMx
0019          real*4          tDivMj


0001    c.....MapMenu.inc
0002
0003    c.....Map menu declarations
0004
0005          integer*2          nMapItems
0006          integer*2          itemGetNewDataSet
0007          integer*2          itemResizeTheMap
0008          integer*2          itemNewMap
0009          integer*2          itemSaveMap
0010          integer*2          itemRedraw
0011          integer*2          itemDone
0012          integer*2          MapMenuID
0013          logical*2          EnableTheItem
0014          logical*2          DisableTheItem
0015          record / MenuHandle / MapMenuHndl
0016
0017    c.....Map menu common block
0018
0019          common    MapMenu    itemGetNewDataSet,    itemResizeTheMap,
0020          &                    itemNewMap,           itemSaveMap,
0021          &                    itemRedraw,           itemDone,
0022          &                    MapMenuID,
0023          &                    EnableTheItem,        DisableTheItem,
0024          &                    MapMenuHndl
0025
0026    c.....Map menu pre-settings
0027
```

27

```
0028        parameter              ( nMapItems            =    6       )
0029        parameter              ( itemGetNewDataSet    =    1       )
0030        parameter              ( itemResizeTheMap     =    2       )
0031        parameter              ( itemNewMap           =    3       )
0032        parameter              ( itemSaveMap          =    4       )
0033        parameter              ( itemRedraw           =    5       )
0034        parameter              ( itemDone             =    6       )
0035        parameter              ( MapMenuID            =  131       )
0036        parameter              ( EnableTheItem        = .true.     )
0037        parameter              ( DisableTheItem       = .false.    )


0001    c.....MBar.inc
0002
0003    c.....MBar declarations
0004
0005        integer*2              MenuBarID
0006        record / MenuHandle / MenuBar
0007
0008    c.....MBar common block
0009
0010        common / MBar      /  MenuBarID,  MenuBar
0011
0012    c.....MBar pre-settings
0013
0014        parameter              ( MenuBarID            = 128 )


0001    C.... NASPCOM.inc:  UNIVERSAL COMMON BLOCK LIST FOR NASP SIM.  07/28/88
0002    C
0003        COMMON /ACOUT1/ CL0_2 ,CD0_2 ,CM0_2 ,CMDE_2,CMQ_2
0004        LOGICAL         PMAN   ,PAUTO ,AUTOP ,RMAN   ,RAUTO ,AUTOR ,
0005        .               SBMAN ,SBAUTO,AUTOSB,HISPED,LOSPED
0006        COMMON /ACSDAT/ PMAN   ,PAUTO ,AUTOP ,RMAN   ,RAUTO ,AUTOR ,
0007        .               SBMAN ,SBAUTO,AUTOSB,HISPED,LOSPED,VDOTC ,
0008        .               ALPNOM,ANZC   ,PHIC   ,PHICG ,VREFG ,ACSQC ,
0009        .               ACSPC ,ACSSBC
0010        COMMON /ACSGAN/ AKVDOT,AKALP ,AK      ,AKNZ   ,AKPHI ,AKSB  ,
0011        .               AKISB ,SBLIMI,SBIN   ,SBOUT ,SBREF
0012        COMMON /ACSOUT/ A01JN1,A02OT1,A03JN1,A04JN1,A05OT1,A06OT1,
0013        .               A07OT1,A08OT1,A09JN1,A10JN1,A11OT1,A31OT1,
0014        .               A32JN1,A33OT1,A34OT1,A61JN1,A62OT1,A63OT1,
0015        .               A64OT1,A65JN1,A66OT1,A67OT1
0016        COMMON /ACTCON/ DELA   ,DELRP ,DELRN ,DELPP ,DELPN ,DELHY ,
0017        .               DERA   ,DERRP ,DERRN ,DERPP ,DERPN ,DERHY ,
0018        .               DR1A   ,DR1RP ,DR1RN ,DR1PP ,DR1PN ,DR1HY ,
0019        .               DR2A   ,DR2RP ,DR2RN ,DR2PP ,DR2PN ,DR2HY ,
0020        .               DRILA ,DRILRP,DRILRN,DRILPP,DRILPN,DRILHY,
0021        .               DRIRA ,DRIRRP,DRIRRN,DRIRPP,DRIRPN,DRIRHY,
0022        .               DROLA ,DROLRP,DROLRN,DROLPP,DROLPN,DROLHY,
0023        .               DRORA ,DRORRP,DRORRN,DRORPP,DRORPN,DRORHY
0024        COMMON /ACTDAT/ DELDAT(10),DERDAT(10),DR1DAT(10),DR2DAT(10),
0025        .               DRILDT(10),DRIRDT(10),DROLDT(10),DRORDT(10)
0026        COMMON /AFILTK/ A64F1K
0027        COMMON /ALGAIN/ GKHDOT,GHLIM ,GKH    ,GKL    ,GAL   ,GLLIM ,
0028        .               GKYDOT,GYMAX ,GKY    ,GFHIMX,GKHEPP
0029        COMMON /ALGANA/ GMCHBP(2),GKYDTA(2),GKYA(2),GYMAXA(2)
0030        COMMON /ALGDAT/ DHDNOM,GAMMA2,GAMMA4,GCX   ,GHF   ,GP     ,
0031        .               GVIC  ,GXK   ,HAL   ,HDBIAS,HDECAY,HDEPP ,
0032        .               HDOTF ,HDOTIC,HDREF ,HEPR   ,HEPRX ,HFF   ,
0033        .               HFINAL,HFLAPE,HMING ,HREF   ,HREF4 ,HTD   ,
0034        .               HWHEEL,IPHASE,RANGE ,SIGMA ,TAU4  ,VGRND ,
0035        .               XAIM2 ,XAIM4 ,XEXP   ,YDOTRW,HDTDCA
```

```
0036          COMMON /ALTFUN/ A       ,RHO    ,G       ,PA     ,TEMPR
0037          COMMON /ARFMIC/ AMSSIC,FMSSIC,FMSMAX,AIXIC ,AIYIC ,AIZIC ,AIXZIC
0038          LOGICAL         IFURUN
0039    C##      INTEGER*8       IACNT ,IBCNT
0040          COMMON /AROFLG/ IFURUN,IACNT ,IBCNT ,INTCNB,ITAERO
0041          COMMON /AROUT2/ CLLB   ,CLLDA ,CLLDR ,CLLP   ,CLLR   ,
0042         .                CLNB   ,CLNDA ,CLNDR ,CLNP   ,CLNR   ,
0043         .                CYB    ,CYDA  ,CYDR
0044          COMMON /ATOVAR/ ITYPE,GS
0045          COMMON /A3OUT1/ CLO_3 ,CDO_3 ,CDI_3 ,CMO_3
0046          COMMON /BLOUT1/ CLO_B ,CDO_B ,CMO_B ,CMDE_B,CMQ_B
0047          COMMON /CGSHFT/ DELX  ,DELY  ,DELZ
0048          CHARACTER*4     DNAMES        ,DNAME ,CMANDS        ,CMAND
0049          COMMON /CHNGDS/ DNAMES(100),DNAME ,CMANDS(20),CMAND ,IDSPGE
0050          COMMON /CICDAT/ ICN    ,RATIO
0051          COMMON /CLCOUT/ CLL    ,CM     ,CLN    ,CD     ,CL     ,CY     ,
0052         .                CN     ,CA     ,XLOD
0053          COMMON /CONCOM/ DAC    ,DEC    ,DRC    ,DELC   ,DERC   ,DR1C   ,
0054         .                DR2C   ,DRILC ,DRIRC ,DROLC ,DRORC ,TBRC
0055          COMMON /CONDAT/ RI1    ,RI2    ,RI3    ,RI4    ,RI5    ,RI6    ,
0056         .                SB     ,SC     ,SM
0057          COMMON /CONFIG/ BODLEN,CGREF ,PMAC   ,ICONFG
0058          LOGICAL         DIRECT
0059          COMMON /CONPOS/ DAP    ,DEP    ,DRP    ,DSBP   ,DBFF   ,DLGP   .
0060         .                THRP   ,DIRECT
0061          COMMON /CONTRL/ DA     ,DE     ,DP     ,DEL    ,DER    ,DP1    ,
0062         .                DR2    ,DRIL   ,DRIR   ,DROL   ,DROR   ,DRI    ,
0063         .                DRO    ,DRV    ,DSB
0064          COMMON /CSINIT/ PO4A   ,PO6A   ,P13A   ,P13B   ,Y05OMG,Y05ZET,
0065         .                Y08A   ,Y10A   ,Y12A
0066          CHARACTER       NAMV*12,NAMI*12,ITYPVR*1
0067          LOGICAL*1       DEGMOD,WGHTMD,NFOUND
0068          COMMON /DACLIS/ VALU(16),IADR(16),NAMV(16),NAMI,DEGMOD(16),
0069         .                WGHTMD(16),ITYPVR(16),NFOUND,XMXVAL(16),
0070         .                XMNVAL(16),SCAL(16),BIAS(16),NBYTES(16)
0071          COMMON /DATAIN/ S      ,B      ,CBAR   ,AMSS   ,AIX    ,AIY    ,
0072         .                AIZ    ,AIXZ   ,AIXE
0073          COMMON /DBANDS/ PDBAND,RDBAND,YDBAND,DEDB1 ,DEDB2 ,DADB1 ,DADB2 ,
0074         .                DRDB1 ,DRDB2
0075          COMMON /DIRGAN/ PDIRGN,RDIRGN,YDIRGN
0076          LOGICAL*1       SSW,OSW
0077          COMMON /DISDAT/ SSW(256),OSW(256)
0078          REAL*8          T      ,P      ,Q      ,R      ,RUH    ,PSIH   ,
0079         .                WH     ,THA    ,PSI    ,PHI    ,DELR   ,LAT_RAD ,
0080         .                LON_RAD
0081    !!!      COMMON /DRVOUT/ T      ,P      ,Q      ,R      ,RUH    ,PSIH   ,
0082    !!!     .                WH     ,THA    ,PSI    ,PHI    ,DELR   ,LAT_RAD ,
0083    !!!     .                LON_RAD ,TDOT   ,PDOT   ,QDOT   ,RDOT   ,RUHDOT,
0084    !!!     .                PSIHDT,WHDOT  ,THADOT,PSIDOT,PHIDOT,DELRDT,
0085    !!!     .                XLATDT,XLNGDT
0086    !!!      COMMON /DRVOT2/ ALP    ,ALPDOT,BTA    ,BTADOT,H      ,HDOT   ,
0087    !!!     .                V      ,VDOT   ,X      ,XDOT   ,Y      ,YDOT   ,VT
0088          REAL*8          VAL
0089          CHARACTER*80    LINE   ,ASCIIN,VARLEQ,VARREQ
0090          COMMON /DSPLAY/ LINE(23),ASCIIN,VARLEQ,VARREQ,VAL    ,IVAL  ,
0091         .                IPAGE   ,INDXVP,IFTP
0092          COMMON /DTABLD/ DTA(128),ATD(128),IDA(128),IAD(128),
0093         .                IDSTAT,IASTAT,NUMDAT,NUMADT
0094          COMMON /ENGCOM/ THRST ,TFACTP
0095          COMMON /ENGCO3/ CTA   ,CTH   ,THRSTH
0096          LOGICAL         ATOPHI
0097          COMMON /ENGDT1/ PHICMD,AC     ,AINF   ,FS     ,EISF   ,SENG   ,
0098         .                XEOUT ,EODRAG,C1COEF,C2COEF,C3COEF,C4COEF,
0099         .                ATOPHI,CAPR   ,EFFISP
```

29

```
0100          COMMON /ENGDT2/ DCMMAX,DCMFCT,DCMEO ,UWEOUT,XNOUT ,CPRS  ,PC    ,
0101          .               DVHCL ,BTAP   ,BTAU   ,BTASD ,BFACTR,TLIMCB,SINDVH,
0102          .               COSDVH
0103          LOGICAL         FOUT  ,NOWC
0104          COMMON /ENGINF/ FQNTY ,FFLOW ,FOUT   ,NOWC
0105          REAL*8          RLOCAL
0106          COMMON /EOMDAT/ RLOCAL,FXB   ,FYB    ,FZB    ,ULV    ,VLV   ,
0107          .               WLV   ,ULVRA ,VLVRA ,WLVRA ,AXLV   ,AYLV  ,
0108          .               AZLV  ,AXH   ,AYH    ,AZH    ,XL1    ,XL2   ,
0109          .               XL3   ,XM1   ,XM2    ,XM3    ,XN1    ,XN2   ,
0110          .               XN3   ,SINPSH,COSPSH
0111          COMMON /EPDATA/ T4T0A (2),T4T0R    ,T4T0    ,T4       ,
0112          .               T5T4A (2),T5T4R    ,T5T4    ,T5       ,
0113          .               P4P0A (2),P4P0R    ,P4P0    ,P4       ,
0114          .               P5P4     ,P5       ,
0115          .               TTEMPR   ,CONRAT   ,WDLAIR   ,
0116          .               WSWPA (2),WSWPR    ,WDSWDP   ,
0117          .               PFPT     ,PTPA     ,PTLSS    ,
0118          .               PCHMBR   ,
0119          .               XMACH (5),PAPSI
0120          COMMON /EPDATH/ COEF1H(2),COEF2H(4),COEF3H(2),COEF4H(4),COEF5H(4),
0121          .               COEF6H(3),COEF7H(7)
0122          COMMON /EPDATL/ COEF1L(2),COEF2L(2),COEF3L(2),COEF4L(2),COEF5L(2),
0123          .               COEF6L(2),COEF7L(2),COEF8L(2),COEF9L(3),COE10L(3),
0124          .               COE11L(3),COE12L(3),COE13L(4),COE14L(2),COE15L(2),
0125          .               COE16L(5)
0126          LOGICAL         ESSWTH
0127          COMMON /ESCOMN/ ESSWTH
0128          LOGICAL         INITLZ,HOLDIC
0129          COMMON /FRSTIC/ INITLZ,HOLDIC
0130          REAL*8          RIC1  ,RIC2   ,RIC3   ,RTG1   ,RTG2   ,RTG3  ,
0131          .               RNGIC ,RNGRW ,CSLTIC,SNLTIC,CSLNIC,SNLNIC,
0132          .               CSLTRW,SNLTRW,CSLNRW,SNLNRW
0133          COMMON /GCCALC/ RIC1  ,RIC2   ,RIC3   ,RTG1   ,RTG2   ,RTG3  ,
0134          .               RNGIC ,RNGRW ,CSLTIC,SNLTIC,CSLNIC,SNLNIC,
0135          .               CSLTRW,SNLTRW,CSLNRW,SNLNRW,BEAR   ,DELAZ
0136          COMMON /GCSOUT/ G01OT1,G02OT1,G03OT1,G04JN1,G05OT1,G06JN1,
0137          .               G07OT1,G08OT1,G09OT1,G10OT1,G11JN1,G12OT1,
0138          .               G31OT1,G32OT1,G33OT1,G34JN1,G35OT1
0139          LOGICAL         GEAR
0140          COMMON /GEARCM/ GEAR
0141          COMMON /GEARFM/ ALGR  ,AMGR  ,ANGR   ,FXGR   ,FYGR   ,FZGR
0142          COMMON /GEARIN/ XG0(3),YG0(3),ZG0(3),XKRG(3),XKD(3),XKBRK,
0143          .               XMUG,XMUR,XMUS(3),AMAXB
0144          LOGICAL         GC
0145          COMMON /GEAROT/ ZG(3),DZG(3),DZGDOT(3),FBG(3),FDG(3),
0146          .               FRG(3),FSG(3),FXBG(3),FYBG(3),GC(3),
0147          .               XGB(3),YGB(3),ZGB(3),PSIW(3)
0148          COMMON /GEREYE/ XGEAR ,ZGEAR ,XEYE   ,ZEYE
0149          COMMON /GFILTK/ G09F1K,G12F1K
0150          LOGICAL         LOGD  ,LOGU
0151          COMMON /GLOBAL04/ ILIST(13,62,3),IVT(3),IDEV(3),IPRI(3),
0152          .                 IOCD(3),IOLIST(4,5,3),ILAST(3),
0153          .                 ITILOC(3),I1553(576),DLK(128),RADAR(16),
0154          .                 IBD(32),LOGD(32),ULK(16),SCV(16),SCB(16),
0155          .                 SCS(16),IBU(32),LOGU(32),PMDU(256)
0156          COMMON /GRANDS/ U
0157          COMMON /GSTINF/ SIGU  ,SIGV  ,SIGW   ,DLU    ,DLV   ,DLW   ,VREF
0158          COMMON /GSTINT/ CU    ,CV    ,CW     ,CF     ,CQ    ,CP    ,
0159          .               EXU1  ,EXV1  ,EXW1   ,EXF1   ,EXQ1  ,EXP1  ,
0160          .               EXV2  ,EXW2  ,EXV3   ,EXW3   ,UG1   ,VG1   ,
0161          .               WG1   ,PG1   ,QG1    ,RG1    ,VG2   ,WG2   ,
0162          .               GU1   ,GV1   ,GW1    ,GP1    ,GQ1   ,GR1   ,
0163          .               GV2   ,GW2   ,SFV
```

30

```
0164            COMMON /GSTOUT/ ALPG  , BTAG  , PG     , QG     , RG     , UG   ,
0165          .                VG    , WG
0166            LOGICAL         GSTSET, GST
0167            COMMON /GSTUSE/ GSTSET, GST
0168            LOGICAL         GINIT
0169            COMMON /GUIDI2/ GDK1  , GDK2  , GDK3  , GDK4  , GDK5  , GDK6  ,
0170          .                GDK7  , GDK8  , GDK9  , GDK10 , GDK11 , GDK12 ,
0171          .                GDK13 , GDK14 , GDK15 , GDK16 , GDK17 , XKROLL,
0172          .                VQ    , XLODM , VS     , GINIT
0173            COMMON /GUIDI3/ RNGBKP(15), AMCHDA(15), QBRNML, QBRNMH
0174            COMMON /GUIDNC/ IGUIDE
0175            COMMON /GUIDO2/ THAACD, PHICD , DD     , THAAC1, PHICD1, DREFP ,
0176          .                ALDREF, HDTREF, DRAGA
0177            COMMON /GUIDO3/ QBRCMD, QBRNOM, QBRERR
0178            CHARACTER*72    HEADER
0179            COMMON /HDRDAT/ HEADER
0180            COMMON /HEATIC/ QTSIC (2), QSIC  (2), TWSIC (2), QTFIC(20),
0181          .                QFIC (20), TWFIC(20)
0182            COMMON /HETDAT/ QTSTAG(2), QSTAG (2), TWSTAG(2), QTFLAT(20),
0183          .                QFLAT(20), TWFLAT(20)
0184            COMMON /HETDOT/ QTSDOT(2), QSDOT (2), TWSDOT(2), QTFDOT(20),
0185          .                QFDOT(20), TWFDOT(20)
0186            COMMON /HETDT1/ XMUINF, RHOLB , ENTHI
0187            COMMON /HETDT2/ ENTHRS(2), ENTHST(2), ENTHWS(2), HTRANS(2),
0188          .                PAST(2)  , PAW(2)   , RHOW(2)  , RSTMST(2),
0189          .                RWMW(2)  , TMPST(2) , TRST (2), VGRAD(2) ,
0190          .                XMUW(2)
0191            COMMON /HETDT3/ SWEEP(2) , COSSWP(2), SINSWP(2), RCURVE(2),
0192          .                IDIMEN(2), HK0(2)   , EMISIV   , RADIAT   ,
0193          .                HCAPS
0194            LOGICAL         TRBLNT
0195            COMMON /HETDT4/ TRBLNT(20), REYNLD(20), ENTHWF(20), ENTHRF(20),
0196          .                TMPRF(20) , ENSTAR(20), TSTAR(20) , HTRANF(20),
0197          .                HIF(20)   , HTF(20)
0198            COMMON /HETDT5/ XDISTN(20), RNREF(20) , ALOGRT(20), COEFM(20),
0199          .                C0(20)    , HCAPF     , C5(20)    , DANGL(20)
0200            COMMON /HTOUT1/ C1    , C2(10), C3     , C4(10), C6     , C7   ,
0201          .                F1(10), F2(10), HK1    , HK2
0202            LOGICAL         DSC
0203            COMMON /ICBOXD/ DSC(6)
0204            COMMON /IDXAC1/ NCA
0205            COMMON /IDXA31/ IAA    , ICA
0206            COMMON /IDXBL1/ MCA
0207            COMMON /IDXTM1/ LAA    , LCA    , LCB    , LCC    , LCD    ,
0208          .                LEA    , LEB    , LEC    , LED
0209            COMMON /IDXT21/ JCA
0210            COMMON /IDXT31/ JAA3   , JAB3   , JAC3   , JAD3   , JCA3   ,
0211          .                JCB3
0212            COMMON /INERTP/ VOI    , ALPOI , BTAOI , GMAOI
0213            COMMON /INGDAT/ NEQN   , HI    , H2
0214            COMMON /INGDA3/ HI3
0215            COMMON /INTDAT/ IH     , IPER  , IERR  , INTTMB, INTBMX, INTTME,
0216          .                INTEMX, INTTMU, INTUMX, INTCNT, INTTBA, INTTEA,
0217          .                INTTUA, INTRCL
0218            COMMON /IN3DAT/ IH3    , IPER3 , IERR3 , IN3TMB, IN3BMX, IN3TME,
0219          .                IN3EMX, IN3TMU, IN3UMX, IN3CNT
0220            COMMON /JETDAT/ XJETF , YJETF , XJETP , YJETP , FRCJET, QBARSW,
0221          .                JETC(12), JET(12)
0222            CHARACTER*8     MCLABL
0223            COMMON /MCDEFL/ MCLABL
0224            CHARACTER*8     MCARRY
0225            COMMON /MCMENA/ INDXMC, MCARRY(96)
0226            COMMON /MCVALU/ PFUEL , XMCMAP, YMCMAP
0227            CHARACTER*12    NAMVM
```

31

```
0228          CHARACTER*1     ITPVRM
0229          LOGICAL*1       DGMODM,WGHMDM,FOUNDM
0230          COMMON /MSCLIS/ VALUM(60),IADRM(60),NAMVM(60),DGMODM(60),
0231      .                   WGHMDM(60),ITPVRM(60),NBYTSM(60),FOUNDM(60),
0232      .                   MSCNUM
0233          CHARACTER*6     MCDEV
0234          LOGICAL         LMSCMP,MCRST ,MCLEAR
0235          COMMON /MSSCMP/ LMSCMP,MCRST ,MCLEAR,MCNT   ,MCNTMX,MCSTAT,
0236      .                   MCDLAY,MCDEV
0237          COMMON /MOMNTS/ AL    ,AM     ,ANN
0238          LOGICAL         LRECV
0239          COMMON /MSSGES/ NUMMES,IRTADR(62),ISUBA(62),IWRDS(62),
0240      .                   INDX(62),LRECV(62)
0241  C##     EXTENDED BASE  /NASPSC/ 356
0242          CHARACTER       LSTNAM*8,LSTTYP*1
0243  crc     INTEGER*1       LSTLEN
0244          INTEGER*2       LSTLEN    !dkh
0245  C##     EXTENDED BLOCK
0246          COMMON
0247      .         /NASPSC/ LSTNAM(3500),LSTTYP(3500),
0248      .                  LSTLEN(3500),LSTADR(3500)
0249  C##     EXTENDED BASE  /NSPACA/ 268
0250  C##     EXTENDED BLOCK
0251          COMMON
0252      .         /NSPACA/ CL0_2A (   189),CD0_2A (   189),CM0_2A (   189),
0253      .                  CMDE_2A(   189),CMQ_2A (   189)
0254  C##     EXTENDED BASE  /NSPAM3/ 292
0255  C##     EXTENDED BLOCK /NSPAM3/
0256          COMMON
0257      .                  CL0_3A (   375),CD0_3A (   375),CDI_3A (    34),
0258      .                  CM0_3A (   375)
0259  C##     EXTENDED BASE  /NSPBLA/ 272
0260  C##     EXTENDED BLOCK
0261          COMMON
0262      .         /NSPBLA/ CL0_BA (   169),CD0_BA (   169),CM0_BA (   169),
0263      .                  CMDE_BA(   169),CMQ_BA (   169)
0264  C##     EXTENDED BASE  /NSPBLP/ 280
0265  C##     EXTENDED BLOCK
0266          COMMON
0267      .         /NSPBLP/ C1ISPA (   810),C2ISPA (   450),C3ISPA (   540),
0268      .                  C4ISPA (   540),CAPR1A (    90),CAPR2A (    50),
0269      .                  CAPR3A (    54),CAPR4A (    54),CDEOA  (    17)
0270  C##     EXTENDED BASE  /NSPDUM/ 256
0271  C##     EXTENDED BLOCK
0272          COMMON
0273      .         /NSPDUM/ ZZZZZZ
0274  C##     EXTENDED BASE  /NSPHET/ 276
0275  C##     EXTENDED BLOCK
0276      .         /NSPHET/ ENTHA  (   192),C1A    (     7),C2A    (    48),
0277      .                  C3A    (     7),C4A    (    48),C6A    (    32),
0278      .                  C7A    (    32),F1A    (    49),F2A    (    49),
0279      .                  HK1A   (     7),HK2A   (     7)
0280          COMMON /ORIGIC/ FIC(13)
0281          COMMON /OVPRES/ BK1    ,BK2    ,BK3    ,BODLEM,CROOTM,CSDIAM,
0282      .                   OVPLNM,OVPRNM,OVPVNM,OVRPRS,PANM   ,PONM  .
0283      .                   XLIFTN,YM
0284          COMMON /PCSOUT/ P01OT1,P02OT1,P03OT1,P04OT1,P05JN1,P06OT1,
0285      .                   P07OT1,P08OT1,P09JN1,P10JN1,P11OT1,P12OT1,
0286      .                   P13OT1,P14OT1,P15JN1,P16OT1,P17OT1,P18OT1
0287          CHARACTER       CPEN*32,NAMVP*12,ITPVRP*1
0288          LOGICAL*1       DGMODP,WGHMDP,FOUNDP
0289          COMMON /PENLIS/ VALUP(12) ,IADRP(12) ,NAMVP(12) ,DGMODP(12),
0290      .                   WGHMDP(12),ITPVRP(12),NBYTSP(12),FOUNDP(12),
0291      .                   CPEN(12)
```

```
0292          COMMON /PFILTK/ P04F1K,P04F2K,P06F1K,P06F2K,P13F1K,P13F2K,
0293       .                  P13F3K,P14F1K
0294          COMMON /PGAINS/ XKDEP ,XKP    ,XKM    ,XKQBAR,XKI
0295          COMMON /PHIDAT/ PHICA(20),PHIMA(20),NUMPHI
0296          LOGICAL         USEPIL,PILTIC
0297          COMMON /PILUSE/ USEPIL,PILTIC
0298          COMMON /PRINFO/ IR     ,OPI    ,TMAX   ,ITITL(12)
0299          COMMON /PTHGAN/ P16KAN,P17KAN,P18KAN,R08KAN,R09KAN,Y21KAN,
0300       .                  Y22KAN,Y23KAN,Y24KAN,Y25KAN
0301          COMMON /QBGOUT/ PHISTB,XLIFTV,XLIFTE,HDOTRF,DLIFT ,XLIFTC,
0302       .                  DTHAC ,QBK1  ,QBK2  ,QBK3  ,QBARRF,ZETA  ,
0303       .                  QBARCA(10),QBARMA(10),NUMQBR
0304          COMMON /RATAC1/ CAM21 ,CALP9
0305          COMMON /RATAD1/ CAL25 ,CAM15 ,CAM34
0306          COMMON /RATBL1/ BALP9 ,BAM13
0307          COMMON /RATTM1/ TALP9 ,TAMA9 ,TAMB5 ,TAMC6 ,TAMD6 ,TAM17 ,TPH10
0308          COMMON /RATT21/ UAM17 ,UPH11
0309          COMMON /RATT31/ UAMC5 ,UAMC2 ,UAMC3 ,UAMC4 ,UAMC6 ,UALP9
0310          COMMON /RCSJET/ RCSN   ,RCSA  ,RCSY   ,RCSM  ,RCSYM ,RCSL
0311          COMMON /RCSOUT/ R01OT1,R02OT1,R03OT1,R04JN1,R05OT1,R06OT1,
0312       .                  R07OT1,R08OT1,R09OT1
0313          REAL*8          REQUAT,RPOLE ,OMEGE ,RADIUS,RADIC ,RADRWY
0314          COMMON /REARTH/ REQUAT,RPOLE ,OMEGE ,RADIUS,RADIC ,RADRWY,
0315       .                  GZERO ,SINLAT,COSLAT,TANLAT
0316          COMMON /REVNUM/ IREVNM
0317          COMMON /RGAINS/ XKDAP ,XKMA   ,XKQBRA
0318          LOGICAL         ONROCK
0319          COMMON /ROCKET/ ONROCK,EXPRAT,RKTHRO,XIMPLS,AREACN,AREATH,
0320       .                  RKTHRS,RKFFLO
0321          LOGICAL         RSTREC,RSTTME
0322          COMMON /RSTTIM/ RSTREC,RSTTME
0323          LOGICAL         OP     ,RST    ,HLD    ,RT     ,ATRM  ,ICEN  ,MDAT
0324          COMMON /RTCDAT/ OP     ,RST    ,HLD    ,RT     ,ATRM  ,ICEN  ,MDAT
0325          CHARACTER*24    RDWPTH,RDRPTH,RDWPTL,RDRPTL
0326          LOGICAL         RTINIT,RTWRIT
0327          COMMON /RTDATA/ RTDATW(174),RTDATR(174),RTINIT,RTWRIT,
0328       .                  NRUNW ,NRUNR ,IFRAMW,IFRAMR,IWRITS,IREADS,
0329       .                  IWRTFR,IREDFR,RDWPTH,RDRPTH,RDWPTL,RDRPTL
0330    C##      COMMON /RT:DIS/ ISSW(8),IOSW(8)
0331    C##      COMMON /RT:INC/ INCERR
0332          COMMON /RTSTAT/ RT1RAT
0333          REAL*8          XLAT0 ,XLNG0 ,RNWLAT,RNWLNG
0334          COMMON /RUNWAY/ XLAT0 ,XLNG0 ,ROTRW ,SINHDR,COSHDR,XDISRW,
0335       .                  YDISRW,XRNWAY,YRNWAY,ROTVIS,SINVIS,COSVIS,
0336       .                  XVIS  ,YVIS   ,ROTMAP,SINMAP,COSMAP,XMAP   ,
0337       .                  YMAP  ,PSIM   ,XVIS0 ,YVIS0 ,RNWLAT,RNWLNG
0338          LOGICAL         LPRNT ,FFEED  ,LLOCK ,UNLOCK,STNDBY,LRUN   ,
0339       .                  CHRTON,LMARK  ,LSCALE,SCINIT
0340          CHARACTER*40    PENBUF
0341          COMMON /SCDATA/ LPRNT ,FFEED  ,LLOCK ,UNLOCK,STNDBY,LRUN   .
0342       .                  CHRTON,LMARK  ,LSCALE,SCINIT,MMPSEC,MMPMIN,
0343       .                  MMPHR ,IRUN   ,PENBUF(16)
0344          CHARACTER*8     SCLABL
0345          COMMON /SCDEFL/ SCLABL
0346          CHARACTER*8     SCARRY
0347          COMMON /SCMENA/ INDXSC,SCARRY(96)
0348          CHARACTER*2     SCTYPE
0349          COMMON /SCTYPE/ SCTYPE
0350          COMMON /SELDAT/ XACCEL, YACCEL, ZACCEL
0351          REAL*8          TIC    ,PIC    ,QIC    ,RIC    ,RUHIC ,PSIHIC,
0352       .                  WHIC   ,THAIC ,PSIIC ,PHIIC ,DELRIC,XLATIC,
0353       .                  XLNGIC
0354          COMMON /SETICS/ TIC    ,PIC    ,QIC    ,RIC    ,RUHIC ,PSIHIC,
0355       .                  WHIC   ,THAIC ,PSIIC ,PHIIC ,DELRIC,XLATIC,
```

```
0356      .                  XLNGIC
0357          COMMON /SETIC2/ VIC   ,ALPIC ,BTAIC ,HIC    ,SALPIC,CALPIC,
0358      .                  SBTAIC,CBTAIC
0359          COMMON /SIGNAL/ SIGNL
0360          COMMON /SIMACC/ AXB   ,AYB   ,AZB   ,ANX   ,ANY   ,ANZ   ,AN
0361          CHARACTER*80    MACFIL,SCRFIL
0362          LOGICAL         LMACRO,SCRIPT
0363          COMMON /SIMCON/ MACFIL,SCRFIL,LMACRO,SCRIPT,LFCM   ,LFCS
0364          COMMON /SIMOUT/ AMCH  ,QBAR  ,GMA    ,DELFP ,UBRA  ,VBRA  ,
0365      .                  WBRA  ,VEAS  ,VCAS
0366          LOGICAL         RUNSTP
0367          COMMON /SIMSTP/ RUNSTP,ALPMAX,ALPMIN,BTAMAX,NFRBLD,HMIN
0368          LOGICAL         NORMAL,LONGIT,LATDIR,LEVEL ,BYPASS,TYPEIN
0369          COMMON /SIMTYP/ NORMAL,LONGIT,LATDIR,LEVEL ,BYPASS,TYPEIN
0370          COMMON /SPEEDB/ SBDRAG,CDSB  ,CDSB1 ,SSB   ,PRCTSB
0371          REAL*8          THATK ,PHITK ,PSITK ,XTK   ,YTK   ,HTK
0372          LOGICAL         PYBM  ,FXRG
0373          COMMON /TARGET/ THATK ,PHITK ,PSITK ,XTK   ,YTK   ,HTK   ,
0374      .                  VEASTK,RANGET,WUTG  ,REVT  ,PSICI ,F1E2  ,
0375      .                  F1K2  ,PYBM  ,TARH  ,TARX  ,TARY  ,FXRG
0376   crc     INTEGER*1       HOMEC ,CURSOR,ENABLC,BLANKC,BLANKL
0377          INTEGER*2       HOMEC ,CURSOR,ENABLC,BLANKC,BLANKL  !dkh
0378          LOGICAL         TWOTEL
0379          COMMON /TERMNL/ ITYCRT     ,TWOTEL     ,HOMEC(10) ,CURSOR(10),
0380      .                  ENABLC(10),BLANKC(10),BLANKL(80),IHOME      ,
0381      .                  ICURS     ,IENABL    ,IBLNKC    ,IBLNKL
0382          COMMON /TMOUT1/ C1ISP ,C2ISP ,C3ISP ,C4ISP ,CAPR1 ,CAPR2 ,CAPR3 ,
0383      .                  CAPR4 ,CDEO
0384          COMMON /TRIGFN/ SINALP,COSALP,SINBTA,COSBTA,SINPHI,COSPHI,
0385      .                  SINPSI,COSPSI,SINTHA,COSTHA
0386          COMMON /TRIMIN/ YTRIM
0387   C##      EXTENDED BASE  /NSPACP/ 288
0388   C##      EXTENDED BLOCK
0389          COMMON
0390      .            /NSPACP/ EISPAA (   187),CAPRAA (   187)
0391          COMMON /T2OUT1/ EISPA ,CAPRA
0392          COMMON /T3DAT1/ CTA1A  (      5),CTA2A  (      2),CTA3A  (      3),
0393      .                  CTA4A  (     54),CTN1A  (      5),CTN2A  (      2),
0394      .                  CTN3A  (      3),CTN4A  (      4),CAPR31A(      5),
0395      .                  CAPR32A(      2),CAPR33A(      3),CAPR34A(     36),
0396      .                  EISP31A(      5),EISP32A(      2),EISP33A(      3),
0397      .                  EISP34A(      4)
0398          COMMON /T3OUT1/ CTA1  ,CTA2  ,CTA3  ,CTA4  ,CTN1  ,CTN2  ,CTN3  ,
0399      .                  CTN4  ,CAPR31,CAPR32,CAPR33,CAPR34,EISP31,EISP32,
0400      .                  EISP33,EISP34
0401          LOGICAL*1       LX
0402          COMMON /VARDAT/ UX(90),LX(50)
0403          LOGICAL*1       LS
0404          COMMON /VARSIM/ US(90),LS(50)
0405          COMMON /VSCDAT/ VBAR (22),TRATIO(22),TPRIME(22),CINF (22),
0406      .                  REYINF    ,TINF      ,SHRAT
0407          LOGICAL         WIND
0408          COMMON /WINDAT/ WIND  ,XWIND ,YWIND ,NUMWND,ALTW(32),VELW(32),
0409      .                  HDGW(32),XWA(32),YWA(32)
0410          COMMON /YCSOUT/ Y01OT1,Y02OT1,Y03OT1,Y04OT1,Y05OT1,Y06OT1,
0411      .                  Y07OT1,Y08OT1,Y09OT1,Y10OT1,Y11OT1,Y12OT1,
0412      .                  Y13OT1,Y14OT1,Y15OT1,Y16OT1,Y17OT1,Y18OT1,
0413      .                  Y19OT1,Y20OT1,Y21OT1,Y22OT1,Y23OT1,Y24OT1,
0414      .                  Y25OT1
0415          COMMON /YFILTK/ Y01F1K,Y05F1K,Y05F2K,Y05F3K,Y05F4K,Y05F5K,
0416      .                  Y08F1K,Y08F2K,Y10F1K,Y10F2K,Y12F1K,Y12F2K
0417          COMMON /YGAINS/ XKRT  ,XKDRP ,XKR   ,XKMR  ,XKQBRR,XKRR
```

34

```
0001    c.....NASPGCom.inc
0002    c
0003    C.... GRAM PROGRAM COMMON BLOCKS.   2/18/89   LJS.
0004    c     SOME GRAM PARAMETER NAMES WERE CHANGED IN THIS ROUTINE
0005    c     TO AVOID CONFLICT WITH DRYDEN SIM PARAMETER NAMES.
0006    c     GRAM PARAMETER NAMES WITHIN THE GRAM ROUTINES WERE UNCHANGED.
0007    c
0008    c     EXTENDED'S REMOVED FOR MAC   10/16/89   LJS.
0009    c
0010    C##      EXTENDED BASE  /C4   / 464
0011    C##      EXTENDED BLOCK /C4   /
0012         COMMON /C4    / GLAT(16),GLON(16),NG    ,P4D(16,26),D4D(16,26),
0013         .              T4D(16,26),SP4(16,26),SD4(16,26),ST4(16,26),
0014         .              THET1 ,THET ,HS
0015         COMMON /CHIC  / LA(4,4),NB(2),IWSYM,UCOEF(14,9),VCOEF(14,9)
0016         COMMON /COMJAC/ XLATJ ,XLONG ,SDA   ,SHA   ,DY    ,R88   ,TE    ,
0017         .              EM
0018         COMMON /COMPER/ SPH   ,SDH   ,STH   ,PRH   ,DRH   ,TRH   ,URH   ,
0019         .              VRH   ,SUH   ,SVH   ,CP88  ,PRHS  ,DRHS  ,TRHS  ,
0020         .              URHS  ,VRHS  ,PRHL  ,DRHL  ,TRHL  ,URHL  ,VRHL  ,
0021         .              SPHS  ,SDHS  ,STHS  ,SUHS  ,SVHS  ,SPHL  ,SDHL  ,
0022         .              STHL  ,SUHL  ,SVHL
0023         COMMON /IOTEMP/ IOTEM1,IOTEM2,IUG   ,IUN   ,DD88  ,XMJD  ,PHI1  ,
0024         .              PHI88 ,NSAME ,RP1   ,RD1   ,RT1   ,SP1   ,SD1   ,
0025         .              ST1   ,RU1   ,RV1   ,SU1   ,SV1   ,MN    ,IDA88 ,
0026         .              IYR   ,H1    ,PHI1R ,THET1R,G88   ,RI    ,H88   ,
0027         .              PHIR  ,THETR ,F10   ,F10B  ,AP    ,IHR   ,MIN   ,
0028         .              NMORE ,DX    ,HL    ,VL    ,DZ    ,B88   ,EPS   ,
0029         .              IOPP  ,LOOK  ,IET   ,GLATX ,RP1S  ,RD1S  ,RT1S  ,
0030         .              RU1S  ,RV1S  ,SP1S  ,SD1S  ,ST1S  ,SU1S  ,SV1S  ,
0031         .              UDS1  ,VDS1  ,UDL1  ,VDL1  ,UDS2  ,VDS2  ,UDL2  ,
0032         .              VDL2, REARTH
0033         COMMON /IPRTP / IPRT
0034    C##      EXTENDED BASE  /PDTCOM/ 448
0035    C##      EXTENDED BLOCK /PDTCOM/
0036         COMMON /PDTCOM/ IU4   ,MONTH ,IOPR  ,PG88(18,19),TG(18,19),
0037         .              DG(18,19),PSP(8,10,12),DSP(8,10,12),TSP(8,10,12),
0038         .              PAQ(17,5),DAQ(17,5),TAQ(17,5),PDQ(17,5),DDQ(17,5),
0039         .              TDQ(17,5),PR(20,10),DR88(20,10),TR(20,10),
0040         .              UAQ(17,5),VAQ(17,5),UDQ(17,5),VDQ(17,5),UR(25,10),
0041         .              VR(25,10),PQ    ,DQ    ,TQ    ,UQ    ,VQ88  ,
0042         .              PQA   ,DQA   ,TQA   ,UA    ,VA    ,IOPQ  ,
0043         .              PLP(25,10),DLP(25,10),TLP(25,10),ULP(25,10),
0044         .              VLP(25,10),UDL(25,10),VDLA(25,10),UDS(25,10),
0045         .              VDSA(25,10)
0046         COMMON /WINCOM/ DH    ,FCORY ,DX5   ,DY5   ,DPX   ,DPY   ,DPXX  ,
0047         .              DPXY  ,DPYY  ,UGH   ,VGH   ,TH    ,DTX   ,DTY   ,
0048         .              DUH   ,DVH   ,PH    ,UPRE  ,VPRE  ,DUPRE ,DVPRE
0049    c
0050    c.... COMMON BLOCKS ADDED IN MODIFYING GRAM AND INTERFACING WITH SIM.
0051    c
0052         COMMON /GRAMOT/ PGH   ,DGH   ,TGH   ,UH    ,VH    ,PS    ,DS    ,
0053         .              TS    ,PGHP  ,DGHP  ,TGHP  ,PHP   ,DHP   ,THP   ,
0054         .              PSH   ,DSH   ,TSH   ,WGH
0055         LOGICAL       GRMATM,G76ATM,GATMP ,GRMWND,GWINDP
0056         COMMON /GRMDAT/ GRMATM,G76ATM,GATMP ,GRMWND,GWINDP,CS76  ,CSU   ,
0057         .              CSP   ,TMPR76,TMPPU ,TMPPP ,PA76  ,PAU   ,PAP   ,
0058         .              RHO76 ,RHOU  ,RHOP  ,UWINDU,UWINDP,VWINDU,VWINDP,
0059         .              USHEAR,VSHEAP
0060    C##      EXTENDED BASE  /NASPGM/ 380
0061    C##      EXTENDED BLOCK /NASPGM/ PDAT(5720) ,DDAT(5720) ,TDAT(5720) ,
0062         COMMON /NASPGM/ PDAT(5720) ,DDAT(5720) ,TDAT(5720) ,
0063         .              SPDAT(5720),SDDAT(5720),STDAT(5720)
```

```
0001     c.....OptFlg.inc
0002
0003     c.....user option flags
0004
0005            integer*2                  oCycle     ! cycle through event loop
0006            integer*2                  oSave      ! save Map as Pict file
0007            integer*2                  oRedraw    ! redraw Map
0008            integer*2                  oNew       ! make new Map
0009            integer*2                  oQuit      ! quit program
0010
0011            common / OptFlg /          oCycle,    oSave,    oRedraw,    oNew,
0012          &                            oQuit
```

```
0001     c.....PenCom.inc
0002
0003     c       common block containing array of pen commands
0004
0005            common / PenCom /                    PenCommand
0006
0007            integer*1               PenCommand(13120)
```

```
0001     c.....PicGrp.inc
0002
0003     c.....Pict group flags
0004
0005            common / PicGrp / PicGroupBeg,PicGroupEnd
0006
0007            integer*2          PicGroupBeg
0008            integer*2          PicGroupEnd
```

```
0001     c.....PlotMenu.inc
0002
0003     c.....Plot menu declarations
0004
0005            integer*2               nPlotItems
0006            integer*2               PlotItemNewPlot
0007            integer*2               PlotItemSavePlot
0008            integer*2               PlotItemRedrawPlot
0009            integer*2               PlotMenuID
0010            record / MenuHandle / PlotMenuHndl
0011
0012     c.....Plot menu common block
0013
0014            common / PlotMenu /    PlotItemNewPlot,         PlotItemSavePlot,
0015          .                        PlotItemRedrawPlot,
0016          .                        PlotMenuID,
0017          .                        PlotMenuHndl
0018
0019     c.....Plot menu pre-settings
0020
0021            parameter               ( nPlotItems         =    3 )
0022            parameter               ( PlotItemNewPlot    =    1 )
0023            parameter               ( PlotItemSavePlot   =    2 )
0024            parameter               ( PlotItemRedrawPlot =    3 )
0025            parameter               ( PlotMenuID         = 131 )
```

```
0001     c.....pntabs.inc
0002
```

```
0003          COMMON / PNTABS /
0004          .  IXABS          , IYABS
0005
0006          INTEGER   IXABS
0007          INTEGER   IYABS


0001   c.....RunSetup.inc
0002
0003   c.....dialog declarations
0004
0005          integer*2          rRunSetupDLOG
0006          integer*2          rCloseButton
0007          integer*2          rRunButton
0008          integer*2          rMapButton
0009          integer*2          rSaveButton
0010          integer*2          rDegWestButton
0011          integer*2          rDegEastButton
0012          integer*2          rMeters
0013          integer*2          rKilometers
0014          integer*2          rSeconds
0015          integer*2          rMinutes
0016          integer*2          rHours
0017          integer*2          rLatitude
0018          integer*2          rLongitude
0019          integer*2          rAltitude
0020          integer*2          rDuration
0021          integer*2          rMissionLabel
0022          integer*2          rWindModelSelector
0023          integer*2          rWindModelSelectorPopup
0024          integer*2          rAscentSelector
0025          integer*2          rAscentSelectorPopup
0026
0027   c.....item ids for dialog-related saved resources
0028
0029          integer*2          rOldMissionText
0030          integer*2          rOldLatitude
0031          integer*2          rOldLongitude
0032          integer*2          rOldDuration
0033          integer*2          rOldAltitude
0034          integer*2          rOldAscent
0035          integer*2          rOldClimate
0036          integer*2          rOldDegRadio
0037          integer*2          rOldDistRadio
0038          integer*2          rOldTimeRadio
0039
0040   c.....working variable declarations
0041
0042          integer*2          DType
0043          integer*2          ItemHit
0044          integer*2          rDegreeSelection
0045          integer*2          rDistanceSelection
0046          integer*2          rTimeSelection
0047          integer*2          ClimateSelection
0048          character*255      AscentSelection
0049
0050          real               xLatitude
0051          real               xLongitude
0052          real               xAltitude
0053          real               xDuration
0054          character*255      xMissionLabel
0055
0056   c.....dialog pre-settings
0057
```

37

```
0058        parameter         ( rRunSetupDLOG          =  256 )
0059        parameter         ( rCloseButton           =    1 )
0060        parameter         ( rRunButton             =    2 )
0061        parameter         ( rMapButton             =    3 )
0062        parameter         ( rSaveButton            =    4 )
0063        parameter         ( rDegWestButton         =    5 )
0064        parameter         ( rDegEastButton         =    6 )
0065        parameter         ( rMeters                =    7 )
0066        parameter         ( rKilometers            =    8 )
0067        parameter         ( rSeconds               =    9 )
0068        parameter         ( rMinutes               =   10 )
0069        parameter         ( rHours                 =   11 )
0070        parameter         ( rMissionLabel          =   19 )
0071        parameter         ( rLatitude              =   20 )
0072        parameter         ( rLongitude             =   21 )
0073        parameter         ( rAltitude              =   22 )
0074        parameter         ( rDuration              =   23 )
0075        parameter         ( rWindModelSelector     =   24 )
0076        parameter         ( rWindModelSelectorPopup =  41 )
0077        parameter         ( rAscentSelector        =   25 )
0078        parameter         ( rAscentSelectorPopup   =   43 )
0079
0080        parameter         ( rOldMissionText        = 1000 )
0081        parameter         ( rOldLatitude           = 1001 )
0082        parameter         ( rOldLongitude          = 1002 )
0083        parameter         ( rOldDuration           = 1003 )
0084        parameter         ( rOldAltitude           = 1004 )
0085        parameter         ( rOldAscent             = 1011 )
0086        parameter         ( rOldClimate            = 1012 )
0087        parameter         ( rOldDegRadio           = 1021 )
0088        parameter         ( rOldDistRadio          = 1022 )
0089        parameter         ( rOldTimeRadio          = 1023 )
0090
0091
0092    c.....dialog-related structures
0093
0094        record / GrafPort       /  SavedPort
0095        record / DialogPtr      /  GetSelection
0096        record / DialogPeek     /  TheDialogPtr
0097        record / TEHandle       /  ThisEditText
0098        record / Handle         /  DItem
0099        record / ControlHandle  /  CItem
0100        record / Rect           /  tempRect
0101
0102    c.....dialog-related common block
0103
0104        common / RunSetup /
0105        &                       SavedPort,
0106        &                       GetSelection,
0107        &                       DType,
0108        &                       DItem,
0109        &                       tempRect,
0110        &                       rDegreeSelection,
0111        &                       rDistanceSelection,
0112        &                       rTimeSelection
0113
0114    c.....trajectory input common block
0115
0116        common / TrajInput
0117        &                       xLatitude,
0118        &                       xLongitude,
0119        &                       xAltitude,
0120        &                       xDuration,
0121        &                       xMissionLabel,
```

38

```
0122          &                   ClimateSelection,
0123          &                   AscentSelection


0001     c.....TicDat.inc
0002
0003     c.....set maximum number major and minor tic marks
0004
0005          parameter          ( nticmx = 20 )
0006
0007     c.....common storage for items relating to tic marks
0008
0009          common / TicDat / xticmj,                 xticmi,
0010          .                 yticmj,                 yticmi,
0011          .                 lticmj,                 lticmi,
0012          .                 ndivmj,
0013          .                 xrefmj,                 yrefmj,
0014          .                 hticmj,                 vticmj,
0015          .                 hticmi,                 vticmi
0016
0017     c.....variable type declarations
0018
0019          integer*2          xticmj
0020          integer*2          xticmi
0021          integer*2          yticmj
0022          integer*2          yticmi
0023          integer*2          lticmj
0024          integer*2          lticmi
0025          integer*2          ndivmj
0026          integer*4          nticmx
0027          real*4             xrefmj(nticmx)
0028          real*4             yrefmj(nticmx)
0029          integer*2          hticmj(nticmx)
0030          integer*2          vticmj(nticmx)
0031          integer*2          hticmi(nticmx,nticmx)
0032          integer*2          vticmi(nticmx,nticmx)


0001     c.....traj.inc
0002
0003     c.....trajectory variables
0004
0005          real*4 Time_Array
0006          real*4 LAT_ARRAY
0007          real*4 LON_ARRAY
0008          real*4 ALT_ARRAY
0009          real*4 GRANGE_ARRAY
0010          real*4 WINDAZ_ARRAY
0011          real*4 WIND_VEL_ARRAY
0012
0013          integer*4 No_Of_Pts
0014
0015     c.....trajectory variable common block
0016
0017          common / traj
0018          &                 Time_Array( 2048 ),
0019          &                 LAT_ARRAY ( 2048 ),
0020          &                 LON_ARRAY ( 2048 ),
0021          &                 ALT_ARRAY ( 2048 ),
0022          &                 GRANGE_ARRAY ( 2048 ),
0023          &                 WINDAZ_ARRAY ( 2048 ),
0024          &                 WIND_VEL_ARRAY ( 2048 ),
0025          &                 No_Of_Pts
```

```
0001    c.....TrjCom.inc
0002
0003    c.....trajectory data common block
0004
0005         integer*2          MaxPts
0006         parameter          (MaxPts=2048)
0007
0008         real*4             TofTab     (MaxPts)
0009         real*4             LngTab     (MaxPts)
0010         real*4             LatTab     (MaxPts)
0011         real*4             AltTab     (MaxPts)
0012
0013         integer*2          JmpTab     (MaxPts)
0014         integer*2          ntrpts
0015
0016         common / TrjCom / TofTab,       LngTab,      LatTab,       AltTab,
0017       &                   JmpTab,       ntrpts
```

```
0001    c.....TrjLim.inc
0002
0003    c.....trajectory data limits
0004
0005         common / TrjLim / MinTof,       MaxTof,      MinLng,       MaxLng,
0006       &                   MinLat,       MaxLat,      MinAlt,       MaxAlt
0007
0008         real*4             MinTof
0009         real*4             MaxTof
0010         real*4             MinLng
0011         real*4             MaxLng
0012         real*4             MinLat
0013         real*4             MaxLat
0014         real*4             MinAlt
0015         real*4             MaxAlt
```

```
0001    c.....VuWind.inc
0002
0003    c.....view window records
0004
0005         common / VuWind / ScrollHndl, TxHndl,    FntDat,     viewRect,
0006       .                   destRect,   TxWptr,    iVuPag,     ScrollPart
0007
0008    c.....scroll bar handle
0009
0010         record / ControlHandle /   ScrollHndl
0011
0012    c.....text edit handle
0013
0014         record / TEHandle /        TxHndl
0015
0016    c.....font characteristics
0017
0018         record . FontInfo          FntDat
0019
0020    c.....Rectangle records
0021
0022         record / rect /            viewRect
0023         record / rect /            destRect
0024
0025    c.....text display window records
```

40

```
0026
0027            record / WindowPtr /            TxWptr
0028
0029     c.....view window page size
0030
0031            integer*2                       iVuPag
0032
0033     c.....scroll part index
0034
0035            integer*2                       ScrollPart
```

```
0001     c.....winlim.inc
0002
0003     c       graphics and Map window limits
0004
0005            common / winlim /
0006            .   igxmin,     igxmax,     igymin,     igymax,
0007            .   ipxmin,     ipxmax,     ipymin,     ipymax,
0008            .   iwxmin,     iwxmax,     iwymin,     iwymax
0009
0010            integer*2    igxmin
0011            integer*2    igxmax
0012            integer*2    igymin
0013            integer*2    igymax
0014            integer*2    ipxmin
0015            integer*2    ipxmax
0016            integer*2    ipymin
0017            integer*2    ipymax
0018            integer*2    iwxmin
0019            integer*2    iwxmax
0020            integer*2    iwymin
0021            integer*2    iwymax
```

10.3 BDPS FORTRAN SOURCE CODE

This section contains a complete listing of the FORTRAN source code which was used in the generation of the BDPS executable program. The source files lines are numbered by the FORTRAN compiler. The majority of the files exist to implement the Macintosh graphical interface and the drift pattern display. Two files, "RunTraj.f" and "Gen4d.f," contain a version of NASA's GRAM which was modified for BDPS. The option for using external forecast data files makes use of the "NOGAPS.f" source file.

```
0001     c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003     !!G toolbox2.finc
0004
0005     c.....Load the ToolBox traps
0006
0007     ''M Inlines.f
0008
0009     c.....Put the following code in the Main segment
0010
0011     !!S Main
0012
0013     c------------------------------------------------------------------------
```

```
Segment Main
0014          subroutine AlertUser
0015    c-------------------------------------------------------------------
0016    c     Display an alert that tells the user an error occurred, then
0017    c     exit the program.
0018
0019    !!SETC USINGINCLUDES = FALSE
0020          implicit none
0021
0022    c.....set up pointer for QuickDraw globals
0023
0024          pointer / QDGlobals /        qdg
0025          common  / QDGPtr /           qdg
0026
0027          integer*2   rUserAlert
0028          integer*2   itemHit
0029
0030          parameter ( rUserAlert=129 )
0031
0032    c-------------------------------------------------------------------
0033
0034          call SetCursor(qdg^.Arrow)
0035          itemHit = Alert(rUserAlert, nil)
0036          call ExitToShell
0037
0038          return
0039          end




0001    c-------------------------------------------------------------------
0002          subroutine AutoScale ( xmin , xmax , ndivmj , dxmin , dxmax ,
0003          .                         xdivmj , xdivmi )
0004    c-------------------------------------------------------------------
0005    c     Compute plot extremes [dxmin] and [dxmax] which will enclose the data
0006    c     extremes [xmin] and [xmax] and yield [ndivmj] divisions [xdivmj] wide,
0007    c     each composed of minor divisions [xdivmi] wide.  The major and minor
0008    c     divisions have only one significant figure each.
0009
0010    c.....external function declaration
0011
0012          external        nquant
0013          integer*2       nquant
0014
0015    c.....if data extremes are equivalent, handle as a special case
0016
0017          if ( xmin.eq.xmax ) then
0018             if ( xmin.ne.0.0 ) then
0019                ilogx  = iint ( alog10(abs(xmin)) - 1.0 )
0020             else
0021                ilogx  = 0
0022             end if
0023             xdivmj = 10.0**ilogx
0024             xdivmi = xdivmj/10.0
0025             dxmin  = xmin - xdivmj
0026             dxmax  = xmin + xdivmj
0027             return
0028          end if
0029
0030    c.....major division width
0031
0032          tmp1   = ( xmax - xmin )/float ( ndivmj )
0033          tmp2   = 10.0**nquant ( alog10(tmp1) , -1.0 )
0034          tmp3   = tmp1/tmp2
0035          if ( tmp3.gt.anint(tmp3) ) then
```

```
0036            xdivmj = tmp2*anint ( tmp3 + 1.0 )
0037          else
0038            xdivmj = tmp2*anint ( tmp3 )
0039          end if
0040
0041    c.....minor division width
0042
0043          xdivmi = 10.0**nquant ( alog10(xdivmj) , -1.0 )
0044          if ( xdivmi.eq.xdivmj ) then
0045            xdivmi = xdivmj/10.0
0046          end if
0047
0048    c.....compute the width of the plot window and the span of the input data
0049
0050          xwidth = xdivmj*float ( ndivmj )
0051          xspan  = xmax - xmin
0052
0053    c.....compute excess width provided with respect to the span of the data
0054    c     extremes and quantize it to the nearest minor division
0055
0056          xces   = xwidth - xspan
0057          nxces  = nquant ( xces , -xdivmi )
0058
0059    c.....allocate half of the excess to the lower end of the plot scale and
0060    c     quantize the minimum plot value to the nearest minor division
0061
0062          nxmin  = nquant ( xmin , -xdivmi )
0063          dxmin  = xdivmi*float ( nxmin - nxces/2 )
0064
0065    c.....offset the maximum plot value from the minimum plot value by the width
0066    c     of the plot window
0067
0068          dxmax  = dxmin + xwidth
0069
0070          return
0071          end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G ToolBox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Main segment
0010
0011    !!S Main
0012
0013    c-------------------------------------------------------------------
Segment Main
0014          program bdps
0015    c-------------------------------------------------------------------
0016    c     Compute and display balloon drift patterns:
0017    c     specifically configured for a Phase 2 SBIP contract with DARPA.
0018    c
0019    c     Developer:   Robert L. Hawkins
0020    c     Date:        May 1991
0021    c     Dialect:     Language Systems FORTRAN 2.1, MPW 3.1
0022    c     Machine:     Macintosh SE/30, 5mb RAM
0023
0024    c....Declare compile-time variable
0025
```

```
0026    !!SETC USINGINCLUDES = FALSE
0027
0028    cc      implicit none
0029
0030    c.....common block definition files
0031
0032         include 'Alert.inc'
0033         include 'AppleMenu.inc'
0034         include 'EditMenu.inc'
0035         include 'FileMenu.inc'
0036         include 'Globals.inc'
0037         include 'MBar.inc'
0038         include 'naspcom.inc'
0039         include 'naspgcom.inc'
0040    c       include 'PlotMenu.inc'
0041         include 'RunSetup.inc'
0042         include 'Traj.inc'
0043
0044         include 'CrvDat.inc'
0045         include 'FntCom.inc'
0046         include 'LatCom.inc'
0047         include 'LngCom.inc'
0048         include 'MapMenu.inc'
0049         include 'OptFlg.inc'
0050         include 'PenCom.inc'
0051         include 'MapCom.inc'
0052         include 'PntAbs.inc'
0053         include 'TicDat.inc'
0054         include 'TrjCom.inc'
0055         include 'VuWind.inc'
0056         include 'WinLim.inc'
0057
0058    c.....set up pointer for QuickDraw globals
0059
0060         pointer / QDGlobals /      qdg
0061         common  / QDGPtr /         qdg
0062         integer*4                  jQDGlobals
0063         external                   jQDGlobals
0064
0065    c--------------------------------------------------------------------------
0066
0067    c.....give us room for memory allocation
0068
0069         call MaxApplZone
0070
0071    c.....call Initialize then unload its segment from memory
0072
0073         call Initialize
0074         call UnloadSeg( %loc(Initialize) )
0075
0076    c.....set up the pointer for QuickDraw globals
0077
0078         qdg  = jQDGlobals()
0079
0080    c.....call eventloop; we will loop forever until user decides to quit
0081
0082         call EventLoop
0083
0084         end


0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
```

```
0004
0005     c.....Load the ToolBox traps
0006
0007     !!M Inlines.f
0008
0009     c.....Put the following code in the Main segment
0010
0011     !!S Main
0012     c-----------------------------------------------------------------------------
Segment Main
0013            subroutine ClearDegreeGroup
0014     c-----------------------------------------------------------------------------
0015
0016     !!SETC USINGINCLUDES = FALSE
0017            implicit none
0018
0019     c.....common block definition files
0020
0021            include 'RunSetup.inc'
0022
0023     c-----------------------------------------------------------------------------
0024
0025     c.....clear the degrees-west button
0026
0027            call GetDItem( %val(GetSelection), %val(rDegWestButton),
0028         &                 %ref(DType), %ref(DItem), %ref(tempRect) )
0029            CItem.CtlH = DItem.bhdl
0030            call SetCtlValue( %val(CItem), %val(0) )
0031
0032     c.....clear the degrees-east button
0033
0034            call GetDItem( %val(GetSelection), %val(rDegEastButton),
0035         &                 %ref(DType), %ref(DItem), %ref(tempRect) )
0036            CItem.CtlH = DItem.bhdl
0037            call SetCtlValue( %val(CItem), %val(0) )
0038
0039            return
0040            end



0001     c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003     !!G toolbox2.finc
0004
0005     c.....Load the ToolBox traps
0006
0007     !!M Inlines.f
0008
0009     c.....Put the following code in the Main segment
0010
0011     !!S Main
0012     c-----------------------------------------------------------------------------
Segment Main
0013            subroutine ClearDistanceGroup
0014     c-----------------------------------------------------------------------------
0015
0016     !!SETC USINGINCLUDES = FALSE
0017            implicit none
0018
0019     c.....common block definition files
0020
0021            include 'RunSetup.inc'
0022
0023     c-----------------------------------------------------------------------------
```

```
0024
0025    c......clear the meters radio button
0026
0027           call GetDItem( %val(GetSelection), %val(rMeters),
0028         &                %ref(DType), %ref(DItem), %ref(tempRect) )
0029           CItem.CtlH = DItem.bhdl
0030           call SetCtlValue( %val(CItem), %val(0) )
0031
0032    c......clear the kilometers radio button
0033
0034           call GetDItem( %val(GetSelection), %val(rKilometers),
0035         &                %ref(DType), %ref(DItem), %ref(tempRect) )
0036           CItem.CtlH = DItem.bhdl
0037           call SetCtlValue( %val(CItem), %val(0) )
0038
0039           return
0040           end


0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Main segment
0010
0011    !!S Main
0012    c----------------------------------------------------------------------------
Segment Main
0013           subroutine ClearTimeGroup
0014    c----------------------------------------------------------------------------
0015
0016    !!SETC USINGINCLUDES = FALSE
0017           implicit none
0018
0019    c.....common block definition files
0020
0021           include 'RunSetup.inc'
0022
0023    c----------------------------------------------------------------------------
0024
0025    c......clear the seconds radio button
0026
0027           call GetDItem( %val(GetSelection), %val(rSeconds),
0028         &                %ref(DType), %ref(DItem), %ref(tempRect) )
0029           CItem.CtlH = DItem.bhdl
0030           call SetCtlValue( %val(CItem), %val(0) )
0031
0032    c......clear the minutes radio button
0033
0034           call GetDItem( %val(GetSelection), %val(rMinutes),
0035         &                %ref(DType), %ref(DItem), %ref(tempRect) )
0036           CItem.CtlH = DItem.bhdl
0037           call SetCtlValue( %val(CItem), %val(0) )
0038
0039    c......clear the hours radio button
0040
0041           call GetDItem( %val(GetSelection), %val(rHours),
0042         &                %ref(DType), %ref(DItem), %ref(tempRect) )
0043           CItem.CtlH = DItem.bhdl
0044           call SetCtlValue( %val(CItem), %val(0) )
```

```
0045
0046            return
0047            end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Main segment
0010
0011    !!S Main
0012    c----------------------------------------------------------------------------
Segment Main
0013            integer*2 function CollectRunInput()
0014    c----------------------------------------------------------------------------
0015    c     Collect all the values entered by the user in the Run Setup dialog
0016
0017    !!SETC USINGINCLUDES = FALSE
0018            implicit none
0019
0020    c.....common block definition files
0021
0022            include 'Globals.inc'
0023            include 'RunSetup.inc'
0024
0025    c.....intermediate text string
0026
0027            string*255 ItemText
0028
0029    c.....character intermediate
0030
0031            character*255 CharData
0032
0033    c----------------------------------------------------------------------------
0034
0035    c.....set the initial function value to zero (no problems)
0036
0037            CollectRunInput = 0
0038
0039    c.....get the initial latitude in degrees north
0040
0041            call GetDItem( %val(GetSelection), %val(rLatitude),
0042          &                 %ref(DType), %ref(DItem), %ref(tempRect) )
0043            call GetIText ( %val(DItem) , %val(ItemText) )
0044            CharData = ItemText
0045            read(CharData,*) xLatitude
0046
0047    c.....check to see that latitude value is ok
0048
0049            if ( xLatitude.lt.-90.0 .or. xLatitude.gt.90.0 ) then
0050              CollectRunInput = rLatitude
0051              return
0052            endif
0053
0054    c.....get the initial longitude in degrees west
0055
0056            call GetDItem( %val(GetSelection), %val(rLongitude),
0057          &                 %ref(DType), %ref(DItem), %ref(tempRect) )
0058            call GetIText ( %val(DItem) , %val(ItemText) )
```

47

```
0059          CharData = ItemText
0060          read(CharData,*) xLongitude
0061
0062    c.....check to see that longitude value is ok
0063
0064          if ( xLongitude.lt.-180.0 .or. xLongitude.gt.180.0 ) then
0065            CollectRunInput = rLongitude
0066            return
0067          endif
0068
0069    c.....convert the longitude value if necessary
0070
0071          if ( rDegreeSelection.eq.rDegWestButton ) then
0072            continue
0073          else if ( rDegreeSelection.eq.rDegEastButton ) then
0074            xLongitude = 360.0 - xLongitude
0075          endif
0076
0077    c.....get the initial altitude in meters
0078
0079          call GetDItem( %val(GetSelection), %val(rAltitude),
0080         &                %ref(DType), %ref(DItem), %ref(tempRect) )
0081          call GetIText ( %val(DItem) , %val(ItemText) )
0082          CharData = ItemText
0083          read(CharData,*) xAltitude
0084
0085    c.....check to see that altitude value is ok
0086
0087          if ( xAltitude.lt.0.0 .or. xAltitude.gt.1000000.0 ) then
0088            CollectRunInput = rAltitude
0089            return
0090          endif
0091
0092    c.....convert the altitude value if necessary
0093
0094          if ( rDistanceSelection.eq.rMeters ) then
0095            continue
0096          else if ( rDistanceSelection.eq.rKilometers ) then
0097            xAltitude = xAltitude *    )0.0
0098          endif
0099
0100    c.....get the flight duration in seconds
0101
0102          call GetDItem( %val(GetSelection), %val(rDuration),
0103         &                %ref(DType), %ref(DItem), %ref(tempRect) )
0104          call GetIText ( %val(DItem) , %val(ItemText) )
0105          CharData = ItemText
0106          read(CharData,*) xDuration
0107
0108    c.....check to see that flight duration value is ok (no more than 30 days)
0109
0110          if ( xDuration.lt.0.0 .or. xDuration.gt.2592000.0 ) then
0111            CollectRunInput = rDuration
0112            return
0113          endif
0114
0115    c.....convert the flight duration value if necessary
0116
0117          if ( rTimeSelection.eq.rSeconds ) then
0118            continue
0119          else if ( rTimeSelection.eq.rMinutes ) then
0120            xDuration = xDuration * 60.0
0121          else if ( rTimeSelection.eq.rHours ) then
0122            xDuration = xDuration * 3600.0
```

```
0123          endif
0124
0125    c.....get the mission label
0126
0127          call GetDItem( %val(GetSelection), %val(rMissionLabel),
0128         &               %ref(DType), %ref(DItem), %ref(tempRect) )
0129          call GetIText ( %val(DItem) , %val(ItemText) )
0130          CharData = ItemText
0131
0132          return
0133          end




0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c-----------------------------------------------------------------------------
0010          subroutine DashIt ( ixold , iyold , ixnew , iynew , icurve )
0011    c-----------------------------------------------------------------------------
0012    c     draw a dashed line between the end points.
0013    c     note:     visible   Ÿ dark  Ÿ pen down ==> iPen = 1
0014    c               invisible Ÿ white Ÿ pen up   ==> iPen = 0
0015
0016          include 'CrvDat.inc'
0017
0018    c.....get Cartesian length components of line segment
0019
0020          ixdif  = ixnew - ixold
0021          iydif  = iynew - iyold
0022
0023    c.....return if segment has zero length
0024
0025          if ( iabs(ixdif).eq.0 .and. iabs(iydif).eq.0 ) then
0026             return
0027          end if
0028
0029    c.....compute unit vector parallel to line segment
0030
0031          r      = sqrt ( float(ixdif)**2 + float(iydif)**2 )
0032          ux     = float(ixdif)/r
0033          uy     = float(iydif)/r
0034
0035    c.....horizontal component exceeds vertical component
0036
0037          if ( iabs(ixdif).ge.iabs(iydif) ) then
0038
0039    c........set loop indices
0040
0041             ix1    = 0
0042             ix2    = iabs(ixdif)
0043
0044    c........loop on vertical scan lines
0045
0046             do i = ix1 , ix2
0047
0048    c...........compute current pixel location
0049
0050                ixref  = ixold + inint ( ux*float(i)/abs(ux) )
0051                iyref  = iyold + inint ( uy*float(i)/abs(ux) )
```

```
0052
0053    c..........increment the pixel counter for the current curve
0054
0055              if ( ixref.ne.ixlast .or. iyref.ne.iylast ) then
0056                  PixCnt(icurve) = 1 + imod ( PixCnt(icurve) , 16 )
0057                  iBitNo          = 16 - PixCnt(icurve)
0058              end if
0059
0060    c..........see if current pixel is visible ( dark ) or invisible ( white )
0061
0062              iPenLst = iPen
0063              iPen    = iibits ( DshMsk(LinTyp(icurve)) , iBitNo , 1 )
0064
0065    c..........save pen down location and draw as necessary
0066
0067              if ( i.eq.ix1 ) then
0068                 if ( iPen.eq.1 ) then
0069                    ixPenU = ixref
0070                    iyPenU = iyref
0071                    ixPenD = ixref
0072                    iyPenD = iyref
0073                 end if
0074              else if ( i.gt.ix1 .and. i.lt.ix2 ) then
0075                 if ( iPen.eq.1 ) then
0076                    ixPenU = ixref
0077                    iyPenU = iyref
0078                    if ( iPenLst.eq.0 ) then
0079                       ixPenD = ixref
0080                       iyPenD = iyref
0081                    end if
0082                 else if ( iPen.eq.0 .and. iPenLst.eq.1 ) then
0083                    call MovAbs ( ixPenD , iyPenD )
0084                    call DrwAbs ( ixPenU , iyPenU )
0085                 end if
0086              else if ( i.eq.ix2 ) then
0087                 if ( iPen.eq.1 )  then
0088                    if ( iPenLst.eq.0 ) then
0089                       call MovAbs ( ixref   , iyref  )
0090                       call DrwAbs ( ixref   , iyref  )
0091                    else if ( iPenLst.eq.1 ) then
0092                       call MovAbs ( ixPenD , iyPenD )
0093                       call DrwAbs ( ixref   , iyref  )
0094                    end if
0095                 end if
0096              end if
0097
0098           end do
0099
0100        end if
0101
0102    c.....vertical component exceeds horizontal component
0103
0104        if ( iabs(ixdif).lt.iabs(iydif) ) then
0105
0106    c.........set loop indices
0107
0108           iy1   = 0
0109           iy2   = iabs(iydif)
0110
0111    c........loop on horizontal scan lines
0112
0113           do i = iy1 , iy2
0114
0115    c...........compute current pixel location
```

```
0116
0117               ixref  = ixold + inint ( ux*float(i)/abs(uy) )
0118               iyref  = iyold + inint ( uy*float(i)/abs(uy) )
0119
0120      c..........increment the pixel counter for the current curve
0121
0122               if ( ixref.ne.ixlast .or. iyref.ne.iylast ) then
0123                  PixCnt(icurve) = 1 + imod ( PixCnt(icurve) , 16 )
0124                  iBitNo         = 16 - PixCnt(icurve)
0125               end if
0126
0127      c..........see if current pixel is visible ( dark ) or invisible ( white )
0128
0129               iPenLst = iPen
0130               iPen    = iibits ( DshMsk(LinTyp(icurve)) , iBitNo , 1 )
0131
0132      c..........save pen down location and draw as necessary
0133
0134               if ( i.eq.iy1 ) then
0135                  if ( iPen.eq.1 ) then
0136                     ixPenU = ixref
0137                     iyPenU = iyref
0138                     ixPenD = ixref
0139                     iyPenD = iyref
0140                  end if
0141               else if ( i.gt.iy1 .and. i.lt.iy2 ) then
0142                  if ( iPen.eq.1 ) then
0143                     ixPenU = ixref
0144                     iyPenU = iyref
0145                     if ( iPenLst.eq.0 ) then
0146                        ixPenD = ixref
0147                        iyPenD = iyref
0148                     end if
0149                  else if ( iPen.eq.0 .and. iPenLst.eq.1 ) then
0150                     call MovAbs ( ixPenD , iyPenD )
0151                     call DrwAbs ( ixPenU , iyPenU )
0152                  end if
0153               else if ( i.eq.iy2 ) then
0154                  if ( iPen.eq.1 )  then
0155                     if ( iPenLst.eq.0 ) then
0156                        call MovAbs ( ixref  , iyref  )
0157                        call DrwAbs ( ixref  , iyref  )
0158                     else if ( iPenLst.eq.1 ) then
0159                        call MovAbs ( ixPenD , iyPenD )
0160                        call DrwAbs ( ixref  , iyref  )
0161                     end if
0162                  end if
0163               end if
0164
0165            end do
0166
0167         end if
0168
0169      c.....save Map end point for reference
0170
0171         ixlast = ixnew
0172         iylast = iynew
0173
0174         return
0175         end




0001      c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
```

51

```
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c------------------------------------------------------------------------
0010          subroutine DisplayMapLimits
0011    c------------------------------------------------------------------------
0012    c     Display the text items containing the map limits
0013
0014    c.....include common block definition files
0015
0016          include 'CrvDat.inc'
0017          include 'DefLim.inc'
0018          include 'FntCom.inc'
0019          include 'MapCom.inc'
0020          include 'MapLim.inc'
0021          include 'TrjLim.inc'
0022
0023    c.....item stuff
0024
0025          record / handle /        ItHndl
0026          record / rect /          ItRect
0027          integer*4                ItType
0028          integer*2                ItNmbr
0029          string*255               ItText
0030
0031    c....."get Map data" dialog interface records
0032
0033          common / MapSetUp  /     MapSetUpPtr,      iGotMapSetUp
0034          record / DialogPtr /     MapSetUpPtr
0035          integer*2                iGotMapSetUp
0036
0037    c.....character strings
0038
0039          character*255            ChrDat
0040
0041    c.....set and select minimum latitude value
0042
0043          ItNmbr = 5
0044          write(ChrDat,*) yMapMn
0045          ItText = ChrDat
0046          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0047        .                 %ref(ItHndl) , %ref(ItRect) )
0048          call SetIText ( %val(ItHndl) , %val(ItText) )
0049          call SelIText ( %val(MapSetUpPtr) , %val(ItNmbr) , %val(0) , %val(32767) )
0050
0051    c.....set maximum latitude value
0052
0053          ItNmbr = 6
0054          write(ChrDat,*) yMapMx
0055          ItText = ChrDat
0056          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0057        .                 %ref(ItHndl) , %ref(ItRect) )
0058          call SetIText ( %val(ItHndl) , %val(ItText) )
0059
0060    c.....set latitude major division size
0061
0062          ItNmbr = 7
0063          write(ChrDat,*) yDivMj
0064          ItText = ChrDat
0065          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0066        .                 %ref(ItHndl) , %ref(ItRect) )
```

52

```
0067          call SetIText ( %val(ItHndl) , %val(ItText) )
0068
0069    c......set latitude minor division size
0070
0071          ItNmbr = 8
0072          write(ChrDat,*) yDivMi
0073          ItText = ChrDat
0074          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0075        .                 %ref(ItHndl) , %ref(ItRect) )
0076          call SetIText ( %val(ItHndl) , %val(ItText) )
0077
0078    c......set minimum longitude value
0079
0080          ItNmbr = 9
0081          write(ChrDat,*) xMapMn
0082          ItText = ChrDat
0083          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0084        .                 %ref(ItHndl) , %ref(ItRect) )
00 5          call SetIText ( %val(ItHndl) , %val(ItText) )
0086
0087    c......set maximum longitude value
0088
0089          ItNmbr = 10
0090          write(ChrDat,*) xMapMx
0091          ItText = ChrDat
0092          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0093        .                 %ref(ItHndl) , %ref(ItRect) )
0094          call SetIText ( %val(ItHndl) , %val(ItText) )
0095
0096    c......set longitude major division size
0097
0098          ItNmbr = 11
0099          write(ChrDat,*) xDivMj
0100          ItText = ChrDat
0101          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0102        .                 %ref(ItHndl) , %ref(ItRect) )
0103          call SetIText ( %val(ItHndl) , %val(ItText) )
0104
0105    c......set longitude minor division size
0106
0107          ItNmbr = 12
0108          write(ChrDat,*) xDivMi
0109          ItText = ChrDat
0110          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0111        .                 %ref(ItHndl) , %ref(ItRect) )
0112          call SetIText ( %val(ItHndl) , %val(ItText) )
0113
0114    c......set time tic increment
0115
0116          if ( TimeTics.eq.1 ) then
0117            write(ChrDat,*) tDivMj
0118          else
0119            ChrDat = ' '
0120          end if
0121          ItNmbr = 13
0122          ItText = ChrDat
0123          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0124        .                 %ref(ItHndl) , %ref(ItRect) )
0125          call SetIText ( %val(ItHndl) , %val(ItText) )
0126
0127          return
0128          end
```

```
0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c-------------------------------------------------------------------------------
0010          subroutine DisplayTimeTics ( xscale , xshift , yscale , yshift )
0011    c-------------------------------------------------------------------------------
0012    c     This subroutine displays time tic marks on the flight path.
0013    c
0014    c     developer:    David F. Smith
0015    c     date:         February 1991
0016
0017          include 'CrvDat.inc'
0018          include 'FntCom.inc'
0019          include 'LatCom.inc'
0020          include 'LngCom.inc'
0021          include 'MapCom.inc'
0022          include 'MapMenu.inc'
0023          include 'MapLim.inc'
0024          include 'OptFlg.inc'
0025          include 'PenCom.inc'
0026          include 'PicGrp.inc'
0027          include 'PntAbs.inc'
0028          include 'TicDat.inc'
0029          include 'TrjCom.inc'
0030          include 'TrjLim.inc'
0031          include 'VuWind.inc'
0032          include 'WinLim.inc'
0033
0034    c.....Rectangle records
0035
0036          record / rect /         bounds
0037
0038    c.....Picture record handle and pointer
0039
0040          common / pict   /       PictHndl
0041          record / PicHandle /    PictHndl
0042          record / PicPtr /       PictPtr
0043
0044    c.....set up pointer for QuickDraw globals
0045
0046          common  / QDGPtr /      QDG
0047          pointer / QDGlobals /   QDG
0048
0049    c.....external function declaration
0050
0051          external                ntrvl
0052          integer*2               ntrvl
0053
0054    c.....graduation output arrays
0055
0056          parameter               ( mxGrad = 101 )
0057          integer*2               mxGrad
0058          real*4                  grVal   (mxGrad)
0059          real*4                  pxRef   (mxGrad)
0060          real*4                  pyRef   (mxGrad)
0061          real*4                  uxRef   (mxGrad)
0062          real*4                  uyRef   (mxGrad)
0063
0064    c.....Intermediate text string
```

```
0065
0066          string*255              TxtOut
0067
0068   c.....character strings
0069
0070          character*255          ChrDat
0071
0072   c.....dialog interface variables ( note that pointers are i*4 )
0073
0074          integer*4              nil
0075
0076   c-------------------------------------------------------------------------
0077   c----------------------- graduate the plot curve -------------------------
0078   c-------------------------------------------------------------------------
0079
0080   c.....calculate graduation values
0081
0082          grVal(1) = tDivMj*anint ( tMapMn/tDivMj )
0083          do while ( grVal(1).lt.tMapMn .or. grVal(1).lt.MinTof )
0084             grVal(1) = tDivMj*anint ( 1.0 + grVal(1)/tDivMj )
0085          end do
0086
0087          i      = 1
0088          do while ( grVal(i).le.tMapMx .and. grVal(i).le.MaxTof )
0089             grVal(i+1) = tDivMj*anint ( 1.0 + grVal(i)/tDivMj )
0090             i       = i + 1
0091          end do
0092          nGrads  = i - 1
0093
0094   c.....draw a tic mark for each graduation value
0095
0096          do i = 1 , nGrads
0097
0098   c........locate the data segment containing the graduation value
0099
0100             j       = ntrvl ( grVal(i) , TofTab , ntrpts , 1 )
0101             k       = j + 1
0102
0103   c........calculate latitude and longitude associated with the time point
0104
0105             tmp1    = grval(i)  - TofTab(j)
0106             tmp2    = TofTab(k) - TofTab(j)
0107             yLat    = LatTab(j) + tmp1*( LatTab(k) - LatTab(j) )/tmp2
0108             if ( JmpTab(j).lt.0 ) then
0109                xLng    = LngTab(j) + tmp1*( LngTab(k) - 360.0 - LngTab(j) )/tmp2
0110                if ( xLng+180.0.lt.0.0 ) then
0111                   xLng    = xLng + 360.0
0112                end if
0113             else if ( JmpTab(k).eq.0 ) then
0114                xLng    = LngTab(j) + tmp1*( LngTab(k) - LngTab(j) )/tmp2
0115             else if ( JmpTab(k).gt.0 ) then
0116                xLng    = LngTab(j) + tmp1*( LngTab(k) + 360.0 - LngTab(j) )/tmp2
0117                if ( xLng-180.0.gt.0.0 ) then
0118                   xLng    = xLng - 360.0
0119                end if
0120             end if
0121
0122   c........compute the window coordinates of the last data point
0123
0124             pxLast = xshift + xscale*LngTab(j)
0125             pyLast = yshift + yscale*LatTab(j)
0126
0127   c........compute the window coordinates of the next data point
0129
```

```
0129                pxNext = xshift + xscale*LngTab(k)
0130                pyNext = yshift + yscale*LatTab(k)
0131
0132     c........compute the window coordinates associated with the tic mark
0133
0134                pxRef(i) = xshift + xscale*xLng
0135                pyRef(i) = yshift + yscale*yLat
0136
0137     c........calculate a unit vector perpendicular to the flight path curve at the
0138     c        tic mark
0139
0140                ux     = pyLast - pyNext
0141                uy     = pxNext - pxLast
0142                rmag   = sqrt ( ux*ux + uy*uy )
0143                if ( rmag.gt.0.0 ) then
0144                   uxRef(i) = ux/rmag
0145                   uyRef(i) = uy/rmag
0146                end if
0147
0148     c........draw the tic mark from one side of the curve to the other
0149
0150                iPxRef = inint(pxRef(i))
0151                iPyRef = inint(pyRef(i))
0152
0153                if ( iPxRef.ge.iPxMin .and. iPxRef.le.iPxMax .and.
0154              .        iPyRef.ge.iPyMin .and. iPyRef.le.iPyMax ) then
0155
0156                   pxout = pxRef(i) - 0.5*uxRef(i)*float(lticmj)
0157                   pyout = pyRef(i) - 0.5*uyRef(i)*float(lticmj)
0158                   call MovAbs ( inint(pxout) , inint(pyout) )
0159
0160                   pxout = pxRef(i) + 0.5*uxRef(i)*float(lticmj)
0161                   pyout = pyRef(i) + 0.5*uyRef(i)*float(lticmj)
0162                   call DrwAbs ( inint(pxout) , inint(pyout) )
0163
0164                end if
0165
0166            end do
0167
0168            call picComment ( %val(picGroupEnd) , %val(0) , %val(nil) )
0169
0170     c.....output and group the graduation values
0171
0172            call picComment ( %val(picGroupBeg) , %val(0) , %val(nil) )
0173            do i = 1 , nGrads
0174               write(ChrDat,1) grVal(i)
0175               call GetStringWidth ( ChrDat , 255 , nchar , iwidth , ixchar )
0176               TxtOut = ChrDat(1:nchar)
0177               pxout  = pxRef(i) - 0.5*uxRef(i)*float(lticmj)
0178              .            - 0.5*float(iWidth)
0179               pyout  = pyRef(i) - 0.5*abs(uyRef(i))*float(lticmj)
0180              .            - float(FontData.ascent+FontData.descent)
0181               iPxRef = inint(pxRef(i))
0182               iPyRef = inint(pyRef(i))
0183               if ( iPxRef.ge.iPxMin .and. iPxRef.le.iPxMax .and.
0184              .        iPyRef.ge.iPyMin .and. iPyRef.le.iPyMax ) then
0185                  call MovAbs ( inint(pxout) , inint(pyout) )
0186                  call DrawString ( %val(TxtOut) )
0187               end if
0188            end do
0189            call picComment ( %val(picGroupEnd) , %val(0) , %val(nil) )
0190
0191          1 format ( f6.1 )
0192
```

```
0193          return
0194          end


0001     c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003     !!G toolbox2.finc
0004
0005     c.....Load the ToolBox traps
0006
0007     !!M Inlines.f
0008
0009     c.....Put the following code in the Main segment
0010
0011     !!S Main
0012
0013     c-----------------------------------------------------------------------
Segment Main
0014          subroutine DoAppleMenu ( menuItem )
0015     c-----------------------------------------------------------------------
0016     c
0017          implicit none
0018     !!SETC USINGINCLUDES = FALSE
0019
0020     c.....include alert definitions, Apple menu definitions
0021
0022          include 'Alert.inc'
0023          include 'AppleMenu.inc'
0024
0025     c.....declare the input argument for the routine
0026
0027          integer*2 menuItem
0028
0029     c.....declare return argument for toolbox "Alert" routine
0030
0031          integer*2 ItemHit
0032
0033     c.....set up variables for handling DAs
0034
0035          string*255 daName
0036          integer*2 daRefNum
0037
0038     c.....set up a structure to save the port while calling DAs
0039
0040          record / GrafPtr / SavePort
0041
0042     c-----------------------------------------------------------------------
0043
0044     c.....use Language System FORTRAN "select case" extension
0045
0046          select case (menuItem)
0047
0048     c........we have selected the About item
0049
0050          case(AppleItemAboutBDPS)
0051
0052     c.........call Alert toolbox function with nil indicating the default filterproc
0053
0054          itemHit = Alert( %val(%AboutAlert), %val(nil) )
0055
0056     c........we must have selected something else (DA, MultiFinder, etc.)
0057
0058          case default
0059
```

```
0060            call GetPort( %ref(SavePort) )
0061            call GetItem( %val(AppleMenuHndl), %val(menuItem), %val(daName) )
0062            daRefNum = OpenDeskAcc( %val(daName) )
0063            call SetPort( %ref(SavePort) )
0064
0065        end select
0066
0067        return
0068        end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Main segment
0010
0011    !!S Main
0012
0013    c-----------------------------------------------------------------------
Segment Main
0014        subroutine DoEditMenu ( menuItem )
0015    c-----------------------------------------------------------------------
0016    c
0017        implicit none
0018    !!SETC USINGINCLUDES = FALSE
0019
0020    c.....include Edit menu definitions
0021
0022        include 'EditMenu.inc'
0023
0024    c.....include global definitions
0025
0026        include 'Globals.inc'
0027
0028    c.....declare the input argument for the routine
0029
0030        integer*2 menuItem
0031
0032        record / DialogPtr / dptr
0033
0034    c-----------------------------------------------------------------------
0035
0036    c.....use Language System FORTRAN "select case" extension
0037
0038        select case (menuItem)
0039
0040          case default
0041
0042        end select
0043
0044        return
0045        end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
```

58

```
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Main segment
0010
0011    !!S Main
0012    c------------------------------------------------------------------------
Segment Main
0013          subroutine DoFileMenu ( menuItem )
0014    c------------------------------------------------------------------------
0015    c
0016          implicit none
0017    !!SETC USINGINCLUDES = FALSE
0018
0019    c.....include definitions
0020
0021          include 'FileInfo.inc'
0022          include 'FileMenu.inc'
0023          include 'Globals.inc'
0024          include 'RunSetup.inc'
0025
0026    c.....declare the input argument for the routine
0027
0028          integer*2              menuItem
0029
0030    c.....prompt string
0031
0032          string*255             Prompt
0033
0034    c.....file information parameters
0035
0036          character*4            FilTyp
0037          character*4            fMaker
0038          string*255             FilNam
0039
0040    c.....I/O error flags
0041
0042          integer*2              ioserr
0043
0044    c------------------------------------------------------------------------
0045
0046    c.....use Language System FORTRAN "select case" extension
0047
0048          select case (menuItem)
0049
0050             case (FileItemNewMission)
0051
0052    c.........first disable the New and Open Mission buttons - one file at a time
0053             call MenuSet( FileMenuID, FileItemNewMission,  .false. )
0054             call MenuSet( FileMenuID, FileItemOpenMission, .false. )
0055    c.........modal dialog for run setup
0056             call RunSetupDialog
0057
0058             case (FileItemOpenMission)
0059
0060    c.........call the open file routine
0061             call SFOpenMissionFile( ioserr, FilTyp, fmaker )
0062             if (ioserr.eq.1) then
0063    c.........disable the New and Open Mission buttons - one file at a time
0064             call MenuSet( FileMenuID, FileItemNewMission,  .false. )
0065             call MenuSet( FileMenuID, FileItemOpenMission, .false. )
0066    c.........set the boolean to indicate that old file is in use
0067                iGotOldFile = .true.
```

```
0068    c...........modal dialog for run setup
0069              call RunSetupDialog
0070    c...........set the boolean to indicate that old file is no longer in use
0071              iGotOldFile = .false.
0072          end if
0073
0074        case (FileItemClose)
0075
0076        case (FileItemQuit)
0077          call ExitToShell
0078
0079      end select
0080
0081      return
0082      end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Main segment
0010
0011    !!S Main
0012
0013    c-------------------------------------------------------------------------
Segment Main
0014        subroutine DoKeyEvent (theEvent)
0015    c-------------------------------------------------------------------------
0016    c    handle a key-press event
0017
0018    !!SETC USINGINCLUDES = FALSE
0019        implicit none
0020
0021    c.....common block definition files
0022
0023        include 'Globals.inc'
0024
0025    c.....character items
0026
0027        integer*2              ChCode
0028        character*1            Ch
0029
0030    c.....menu pieces (menuResult = merge of menuID and menuItem)
0031
0032        integer*4 menuResult
0033        integer*4 itemMask /32767/
0034        integer*4 menuMask /-16/
0035        integer*2 menuID
0036        integer*2 menuItem
0037
0038        record /EventRecord/          theEvent
0039
0040    c-------------------------------------------------------------------------
0041
0042    c.....get the character
0043
0044        ChCode = jiand ( theEvent.message , CharCodeMask )
0045
0046    c.....get the character's ASCII representation
```

```
0047
0048          ch      = char ( ChCode )
0049
0050    c.....check for the Cmd key depression
0051
0052          ChCode = jiand ( theEvent.modifiers , CmdKey )
0053
0054          if ( ChCode.ne.0 ) then
0055
0056             menuResult = MenuKey( %val(ch) )
0057
0058    c.......extract the menu and item numbers from within menuResult
0059
0060             menuItem = jiand ( menuResult, itemMask )
0061             menuID   = jishft( menuResult, menuMask )
0062
0063    c.......call my Menu handler with the selection
0064
0065             if (menuID .ne. 0) call DoMenu (menuID, menuItem)
0066
0067    c.......handle any cut, copy, or paste
0068             if ( (ch.eq.'x' .or. ch.eq.'X') .and. theInput .ne. nil )
0069         &     call TECut  ( %ref(theInput) )
0070             if ( (ch.eq.'c' .or. ch.eq.'C') .and. theInput .ne. nil )
0071         &     call TECopy ( %ref(theInput) )
0072             if ( (ch.eq.'v' .or. ch.eq.'V') .and. theInput .ne. nil )
0073         &     call TEPaste( %ref(theInput) )
0074
0075    c.....if CmdKey not depressed, then it must be TE
0076
0077           else if ( theInput .ne. nil ) then
0078
0079             call TEKey ( %val(ch), %ref(theInput) )
0080
0081          endif
0082
0083          return
0084          end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c-----------------------------------------------------------------------------
0010          subroutine DoMapMenu ( TheEvent , Option )
0011    c-----------------------------------------------------------------------------
0012
0013    c.....include files
0014
0015          include 'MapMenu.inc'
0016          include 'OptFlg.inc'
0017
0018    c.....event record
0019
0020          record  EventRecord      TheEvent
0021
0022    c.....user option index
0023
0024          integer*2                Option
```

61

```
0025
0026      c.....LongWord containing menu selection parameters
0027
0028          integer*4                MenuChoice
0029
0030      c.....LongWord masks
0031
0032          integer*4                ItemMask
0033          integer*4                MenuMask
0034
0035      c.....character items
0036
0037          integer*2                ChCode
0038          character*1              Ch
0039
0040      c.....set local integer masks
0041
0042          data ItemMask            / 32767 /
0043          data MenuMask            /   -16 /
0044
0045      c-------------------------------------------------------------------------
0046
0047      c.....handle MouseDown in menuBar
0048
0049          if ( TheEvent.what.eq.mouseDown ) then
0050              MenuChoice = MenuSelect ( %val(TheEvent.where) )
0051          end if
0052
0053      c.....handle keyDown events
0054
0055          if ( TheEvent.what.eq.keyDown ) then
0056              chcode = jiand ( TheEvent.message , CharCodeMask )
0057              ch     = char ( chcode )
0058
0059      c........Enter or Return: Make a new Map
0060
0061              if ( chcode.eq.3 .or. chcode.eq.13 ) then
0062                  call HiliteMenu ( %val(MapMenuID) )
0063                  menu       = MapMenuID
0064                  item       = itemNewMap
0065                  MenuChoice = item + jishft ( menu , - MenuMask )
0066
0067      c........If command key was depressed, activate equivalent menu item
0068
0069              else
0070                  chcode = jiand ( TheEvent.modifiers , CmdKey )
0071                  if ( chcode.ne.0 ) then
0072                      MenuChoice = MenuKey ( %val(ch) )
0073
0074      c........Otnerwise perform no operation
0075
0076                  else
0077                      MenuChoice = 0
0078                  end if
0079              end if
0080          end if
0081
0082      c.....determine which menu and menu item were selected
0083
0084          if ( MenuChoice.gt.0 ) then
0085              item   = jiand ( MenuChoice , ItemMask )
0086              menu   = jishft ( MenuChoice , MenuMask )
0087
0088      c........Options menu
```

```
0089
0090              if ( menu.eq.MapMenuID ) then
0091
0092   c...........get a new data set to Map
0093
0094                  if ( item.eq.itemGetNewDataSet ) then
0095                     call ReadFlightData
0096                     Option = oCycle
0097                  end if
0098
0099   c..........resize the map window
0100
0101                  if ( item.eq.itemResizeTheMap ) then
0102                     call ResizeTheMap
0103                     Option = oCycle
0104                  end if
0105
0106   c...........make new Map
0107
0108                  if ( item.eq.itemNewMap ) then
0109                     Option = oNew
0110                  end if
0111
0112   c...........save Map into Pict file and redraw the Map window
0113
0114                  if ( item.eq.itemSaveMap ) then
0115                     call SaveTheMap
0116                     Option = oCycle
0117                  end if
0118
0119   c...........redraw the Map window
0120
0121                  if ( item.eq.itemRedraw ) then
0122                     Option = oRedraw
0123                  end if
0124
0125   c...........quit from the program
0126
0127                  if ( item.eq.itemDone ) then
0128                     Option = oQuit
0129                  end if
0130
0131              end if
0132
0133          end if
0134
0135   c.....turn off all menu highlighting
0136
0137       call HiliteMenu ( 0 )
0138
0139       return
0140       end



0001   c......Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003   !!G toolbox2.finc
0004
0005   c.....Load the ToolBox traps
0006
0007   !!M Inlines.f
0008
0009   c.....Put the following code in the Main segment
0010
```

```
0011     !!S Main
0012
0013     c-----------------------------------------------------------------------
Segment Main
0014           subroutine DoMenu ( menuID, menuItem )
0015     c-----------------------------------------------------------------------
0016     c
0017           implicit none
0018     !!SETC USINGINCLUDES = FALSE
0019
0020     c.....common block definition files
0021
0022           include 'AppleMenu.inc'
0023           include 'FileMenu.inc'
0024           include 'EditMenu.inc'
0025           include 'MapMenu.inc'
0026           include 'MBar.inc'
0027           include 'Globals.inc'
0028
0029     c.....menu pieces passed in as arguments
0030
0031           integer*2 menuID
0032           integer*2 menuItem
0033
0034     c-----------------------------------------------------------------------
0035
0036     c.....use Language System FORTRAN "select case" extension
0037
0038           select case (menuID)
0039
0040     c.......choice from Apple menu
0041
0042             case(AppleMenuID)
0043
0044               call DoAppleMenu ( menuItem )
0045
0046     c.......choice from File menu
0047
0048             case(FileMenuID)
0049
0050               call DoFileMenu ( menuItem )
0051
0052     c.......choice from Edit menu
0053
0054             case(EditMenuID)
0055
0056               call DoEditMenu ( menuItem )
0057
0058     c.......choice from Plot menu
0059
0060             case(MapMenuID)
0061
0062               call DoMapMenu ( menuItem )
0063
0064           end select
0065
0066     c.....turn off high-lighting for the selected menu
0067
0068           call HiliteMenu( %val(0) )
0069
0070           return
0071           end
```

```
0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Main segment
0010
0011    !!S Main
0012
0013    c-----------------------------------------------------------------------
Segment Main
0014          subroutine DoPlotMenu ( menuItem )
0015    c------------------------------------    ------------------------------------
0016    c
0017          implicit none
0018    !!SETC USINGINCLUDES = FALSE
0019
0020    c.....include Plot menu definitions
0021
0022          include 'PlotMenu.inc'
0023
0024    c.....include global definitions
0025
0026          include 'Globals.inc'
0027
0028    c.....declare the input argument for the routine
0029
0030          integer*2 menuItem
0031
0032          record / DialogPtr / dptr
0033
0034    c-----------------------------------------------------------------------
0035
0036    c.....use Language System FORTRAN "select case" extension
0037
0038          select case (menuItem)
0039
0040            case default
0041
0042          end select
0043
0044          return
0045          end
```

```
0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    ''M Inlines.f
0008
0009    c-----------------------------------------------------------------------
0010          subroutine DrawFlightPath ( xscale , xshift , yscale , yshift )
0011    c-----------------------------------------------------------------------
0012    c     This subroutine overlays a flight path over the map.
0013    c
0014    c     developer:   David F. Smith
0015    c     date:        February 1991
0016
```

```
0017          include 'CrvDat.inc'
0018          include 'FntCom.inc'
0019          include 'LatCom.inc'
0020          include 'LngCom.inc'
0021          include 'MapCom.inc'
0022          include 'MapMenu.inc'
0023          include 'MapLim.inc'
0024          include 'OptFlg.inc'
0025          include 'PenCom.inc'
0026          include 'PicGrp.inc'
0027          include 'PntAbs.inc'
0028          include 'TicDat.inc'
0029          include 'TrjCom.inc'
0030          include 'VuWind.inc'
0031          include 'WinLim.inc'
0032
0033    c.....Rectangle records
0034
0035          record / rect /            bounds
0036
0037    c.....Picture record handle and pointer
0038
0039          common / pict   /          PictHndl
0040          record / PicHandle /       PictHndl
0041          record / PicPtr /          PictPtr
0042
0043    c.....set up pointer for QuickDraw globals
0044
0045          common  / QDGPtr /         QDG
0046          pointer / QDGlobals /      QDG
0047
0048    c.....set pen size for flight path
0049
0050          call PenSize ( %val(2) , %val(2) )
0051
0052    c.....group the flight path data points
0053
0054          call picComment ( %val(picGroupBeg) , %val(0) , %val(nil) )
0055          do i = 1 , ntrpts
0056
0057    c........1st point
0058
0059             if ( i.eq.1 ) then
0060                pxout = xshift + xscale*LngTab(i)
0061                pyout = yshift + yscale*LatTab(i)
0062                call MovAbs ( inint(pxout) , inint(pyout) )
0063
0064    c........all other points
0065
0066             else if ( i.ne.1 ) then
0067
0068    c...........a longitude discontinuity occurred in the Westward direction
0069
0070                if ( JmpTab(i).lt.0 ) then
0071                   pxout = xshift + xscale*( LngTab(i) - 360.0 )
0072                   pyout = yshift + yscale*LatTab(i)
0073                   call DshAbs ( inint(pxout) , inint(pyout) , icurve )
0074                   pxout = xshift + xscale*( LngTab(i-1) + 360.0 )
0075                   pyout = yshift + yscale*LatTab(i-1)
0076                   call MovAbs ( inint(pxout) , inint(pyout) )
0077                   pxout = xshift + xscale*LngTab(i)
0078                   pyout = yshift + yscale*LatTab(i)
0079                   call DshAbs ( inint(pxout) , inint(pyout) , icurve )
0080
```

```
0081    c............o longitude discontinuity occurred
0082
0083                else if ( JmpTab(i).eq.0 ) then
0084                    pxout  = xshift + xscale*LngTab(i)
0085                    pyout  = yshift + yscale*LatTab(i)
0086                    call DshAbs ( inint(pxout) , inint(pyout) , icurve )
0087
0088    c...........a longitude discontinuity occurred in the Eastward direction
0089
0090                else if ( JmpTab(i).gt.0 ) then
0091                    pxout  = xshift + xscale*( LngTab(i) + 360.0 )
0092                    pyout  = yshift + yscale*LatTab(i)
0093                    call DshAbs ( inint(pxout) , inint(pyout) , icurve )
0094                    pxout  = xshift + xscale*( LngTab(i-1) - 360.0 )
0095                    pyout  = yshift + yscale*LatTab(i-1)
0096                    call MovAbs ( inint(pxout) , inint(pyout) )
0097                    pxout  = xshift + xscale*LngTab(i)
0098                    pyout  = yshift + yscale*LatTab(i)
0099                    call DshAbs ( inint(pxout) , inint(pyout) , icurve )
0100                end if
0101            end if
0102        end do
0103        call picComment ( %val(picGroupEnd) , %val(0) , %val(nil) )
0104
0105    c.....reset pen size
0106
0107        call PenSize ( %val(1) , %val(1) )
0108
0109    c.....display time tic marks
0110
0111        if ( TimeTics .and. ntrpts.gt.1 ) then
0112            call DisplayTimeTics ( xscale , xshift , yscale , yshift )
0113        end if
0114
0115        return
0116        end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    !!S DrawTheMap
0010    ~------------------------------------------------------------------------
0011        subroutine DrawTheMap ( Option )
0012    c------------------------------------------------------------------------
0013    c      This subroutine draws the global map data in the graphics window and
0014    c      overlays a flight path over it.
0015    c
0016    c      developer:    David F. Smith
0017    c      date:         February 1991
0018
0019        include 'CrvDat.inc'
0020        include 'FntCom.inc'
0021        include 'LatCom.inc'
0022        include 'LngCom.inc'
0023        include 'MapCom.inc'
0024        include 'MapMenu.inc'
0025        include 'MapLim.inc'
0026        include 'OptFlg.inc'
```

```
0027          include 'PenCom.inc'
0028          include 'PicGrp.inc'
0029          include 'PntAbs.inc'
0030          include 'TicDat.inc'
0031          include 'TrjCom.inc'
0032          include 'VuWind.inc'
0033          include 'WinLim.inc'
0034
0035    c.....Rectangle records
0036
0037          record / rect /           bounds
0038
0039    c.....graphics window records
0040
0041          common / MapWindow /      MapWPtr
0042          record / WindowPtr /      MapWptr
0043
0044    c.....Map window title
0045
0046          string*255                MapWTitle
0047
0048    c.....Picture record handle and pointer
0049
0050          common / pict    /        PictHndl
0051          record / PicHandle /      PictHndl
0052          record / PicPtr /         PictPtr
0053
0054    c.....set up pointer for QuickDraw globals
0055
0056          common  / QDGPtr /        QDG
0057          pointer / QDGlobals /     QDG
0058          integer*4                 jQDGlobals
0059          external                  jQDGlobals
0060
0061    c.....Intermediate text string
0062
0063          string*255                TxtOut
0064
0065    c.....character strings
0066
0067          character*255             ChrDat
0068          character*9               cscale
0069          character*9               cshift
0070          character*1               onechr
0071
0072    c.....dialog interface variables ( note that pointers are i*4 )
0073
0074          integer*4                 nil
0075
0076    c.....define logical*1 items to emulate Boolean Pascal items
0077
0078          logical*1                 visibl
0079          logical*1                 goAway
0080          logical*1                 fUpdate
0081
0082    c.....temporary table of x window coordinates for character placement
0083
0084          integer*2                 ixchar(255)
0085
0086    c.....user option flag
0087
0088          integer*2                 Option
0089
0090    c.....calculate graph window limits
```

```
0091
0092            iGxMax = iGxMin + iint ( MapHRes*MapWidth  )
0093            iGyMax = iGyMin + iint ( MapVRes*MapHeight )
0094
0095    c.....set draw window limits
0096
0097            iWxMin = iGxMin + 35
0098            iWxMax = iGxMax - 15
0099            iWyMin = iGyMin + 20
0100            iWyMax = iGyMax - 24
0101
0102    c.....create a visible graphics window with no goAway box and an invisible
0103    c       analog for refreshing
0104
0105            if ( iGotMapWptr.eq.0 ) then
0106               bounds.left   = iGxMin
0107               bounds.right  = iGxMax
0108               bounds.top    = iGyMin + 38
0109               bounds.bottom = iGyMax + 38
0110               visibl        = .true.
0111               goAway        = .false.
0112               MapWTitle     = 'Map Window'
0113               MapWptr       = NewWindow ( %val(nil) , %val(bounds) ,
0114               .                           %val(MapWTitle) , %val(visibl) ,
0115               .                           %val(noGrowDocProc) , %val(int4(-1)) ,
0116               .                           %val(goAway) , %val(nil) )
0117               iGotMapWptr   = 1
0118            else
0119               fUpdate = .true.
0120               mWidth  = iGxMax - iGxMin
0121               mHeight = iGyMax - iGyMin
0122               call SizeWindow  ( %val(MapWptr) , %val(mWidth) , %val(mHeight) ,
0123               .                  %val(fUpdate) )
0124            end if
0125
0126    c.....incorporate the entire window into the update region
0127
0128            call InvalRect ( %val(MapWPtr.WP^.portRect) )
0129
0130    c.....initiate Plot window update
0131
0132            call BeginUpdate ( %val(MapWptr) )
0133
0134    c.....open drawing port
0135
0136            call SetPort ( %val(MapWptr) )
0137            call ShowWindow ( %val(MapWptr) )
0138
0139    c.....set clip and visible region boundaries to port rectangle
0140
0141            MapWPtr.WP^.clipRgn.RgnH^.RgnP^.RgnBBox = MapWPtr.WP^.portRect
0142            MapWPtr.WP^.visRgn.RgnH^.RgnP^.RgnBBox  = MapWPtr.WP^.portRect
0143
0144    c.....erase the visible region
0145
0146            call EraseRect ( %val(MapWPtr.WP^.portRect) )
0147
0148    c.....Open picture in case user wishes to save Map
0149
0150            PictHndl = OpenPicture ( %val(MapWPtr.WP^.portRect) )
0151            call ShowPen
0152
0153    c.....set font size to Geneva 9 and get font information
0154
```

```
0155          FntNam = 'Geneva'
0156          call GetFNum ( %val(FntNam) , FntNum )
0157          call TextFont ( %val(FntNum) )
0158          call TextSize ( %val(9) )
0159          call GetFontInfo ( %ref(FontData) )
0160
0161  c.....Set initial pointer to bottom left corner of Map window
0162
0163          ixabs  = iWxMin
0164          iyabs  = iWyMin
0165
0166  c.....set pen size for drawing map
0167
0168          call PenSize ( %val(1) , %val(1) )
0169
0170  c.....set Map window screen limits
0171
0172          ipxmin = iWxMin
0173          ipxmax = iWxMax
0174          ipymin = iWyMin
0175          ipymax = iWyMax
0176
0177  c.....determine spans of Map data
0178
0179          xmSpan   = xMapMx - xMapMn
0180          ymSpan   = yMapMx - yMapMn
0181
0182  c.....determine number of major and minor tic marks
0183
0184          xTicMj = inint ( xmSpan/xDivMj )
0185          xTicMi = inint ( xDivMj/xDivMi )
0186          yTicMj = inint ( ymSpan/yDivMj )
0187          yTicMi = inint ( yDivMj/yDivMi )
0188
0189  c.....determine scale factors which will scale the data to fill the
0190  c       Map window
0191
0192          xscale = float ( ipxmax - ipxmin )/xmSpan
0193          yscale = float ( ipymax - ipymin )/ymSpan
0194
0195  c.....determine offsets which will shift the scaled data into the Map
0196  c       window
0197
0198          xshift = float(ipxmin) - xscale*xMapMn
0199          yshift = float(ipymin) - yscale*yMapMn
0200
0201  c.....loop on data points to draw curve and make sure it is grouped
0202
0203          call picComment ( %val(picGroupBeg) , %val(0) , %val(nil) )
0204
0205          do i = 1 , 13120
0206
0207  c........determine Map window coordinates of data point
0208
0209             pxout  = xshift + xscale*Longitude(i)
0210             pyout  = yshift + yscale*Latitude(i)
0211
0212  c........either draw a line segment or position the Map pointer
0213
0214             if ( PenCommand(i).eq.0 ) then
0215                call MovAbs ( inint(pxout) , inint(pyout) )
0216             else if ( PenCommand(i).eq.1 ) then
0217                call DshAbs ( inint(pxout) , inint(pyout) , icurve )
0218             end if
```

```
0219
0220          end do
0221          call picComment ( %val(picGroupEnd) , %val(0) , %val(nil) )
0222
0223   c.....form the Map window frame
0224
0225          call picComment ( %val(picGroupBeg) , %val(0) , %val(nil) )
0226          if ( iMapyp.eq.1 .or. icurve.eq.ncurve ) then
0227             call MovAbs ( ipxmin , ipymin )
0228             call DrwAbs ( ipxmax , ipymin )
0229             call DrwAbs ( ipxmax , ipymax )
0230             call DrwAbs ( ipxmin , ipymax )
0231             call DrwAbs ( ipxmin , ipymin )
0232          end if
0233          call picComment ( %val(picGroupEnd) , %val(0) , %val(nil) )
0234
0235   c.....draw x axis major tic marks or vertical grid lines and calculate
0236   c       associated reference values
0237
0238          call picComment ( %val(picGroupBeg) , %val(0) , %val(nil) )
0239          do i = 1 , xTicMj + 1
0240             xrefmj(i) = xMapMn + xDivMj*float(i-1)
0241             hticmj(i) = inint ( xshift + xscale*xrefmj(i) )
0242             if ( GridLines.eq.1 ) then
0243                if ( i.gt.1 .and. i.le.xTicMj ) then
0244                call MovAbs ( hticmj(i) , ipymin )
0245                call DrwAbs ( hticmj(i) , ipymax )
0246                end if
0247             else
0248                call MovAbs ( hticmj(i) , ipymin          )
0249                call DrwAbs ( hticmj(i) , ipymin + lticmj )
0250                call MovAbs ( hticmj(i) , ipymax - lticmj )
0251                call DrwAbs ( hticmj(i) , ipymax          )
0252             end if
0253          end do
0254          call picComment ( %val(picGroupEnd) , %val(0) , %val(nil) )
0255
0256   c.....draw y axis major tic marks or horizontal grid lines and calculate
0257   c       associated reference values
0258
0259          call picComment ( %val(picGroupBeg) , %val(0) , %val(nil) )
0260          do i = 1 , yTicMj + 1
0261             yrefmj(i) = yMapMn + yDivMj*float(i-1)
0262             vticmj(i) = inint ( yshift + yscale*yrefmj(i) )
0263             if ( GridLines.eq.1 ) then
0264                if ( i.gt.1 .and. i.le.yTicMj ) then
0265                call MovAbs ( ipxmin , vticmj(i) )
0266                call DrwAbs ( ipxmax , vticmj(i) )
0267                end if
0268             else
0269                call MovAbs ( ipxmin          , vticmj(i) )
0270                call DrwAbs ( ipxmin + lticmj , vticmj(i) )
0271                call MovAbs ( ipxmax - lticmj , vticmj(i) )
0272                call DrwAbs ( ipxmax          , vticmj(i) )
0273             end if
0274          end do
0275          call picComment ( %val(picGroupEnd) , %val(0) , %val(nil) )
0276
0277   c.....draw each of the x axis minor tic marks
0278
0279          call picComment ( %val(picGroupBeg) , %val(0) , %val(nil) )
0280          do i = 1 , xTicMj
0281             do j = 1 , xTicMi - 1
0282                xrefmi    = xrefmj(i) + xDivMi*float(j)
```

71

```
0283                    hticmi(i,j) = inint ( xshift + xscale*xrefmi )
0284                    call MovAbs ( hticmi(i,j) , ipymin              )
0285                    call DrwAbs ( hticmi(i,j) , ipymin + lticmi )
0286                    call MovAbs ( hticmi(i,j) , ipymax - lticmi )
0287                    call DrwAbs ( hticmi(i,j) , ipymax              )
0288                end do
0289            end do
0290            call picComment ( %val(picGroupEnd) , %val(0) , %val(nil) )
0291
0292    c.....draw each of the y axis minor tic marks
0293
0294            call picComment ( %val(picGroupBeg) , %val(0) , %val(nil) )
0295            do i = 1 , yTicMj
0296                do j = 1 , yTicMi - 1
0297                    yrefmi      = yrefmj(i) + yDivMi*float(j)
0298                    vticmi(i,j) = inint ( yshift + yscale*yrefmi )
0299                    call MovAbs ( ipxmin          , vticmi(i,j) )
0300                    call DrwAbs ( ipxmin + lticmi , vticmi(i,j) )
0301                    call MovAbs ( ipxmax - lticmi , vticmi(i,j) )
0302                    call DrwAbs ( ipxmax          , vticmi(i,j) )
0303                end do
0304            end do
0305            call picComment ( %val(picGroupEnd) , %val(0) , %val(nil) )
0306
0307    c.....output reference values associated with x axis major tic marks
0308
0309            call picComment ( %val(picGroupBeg) , %val(0) , %val(nil) )
0310            do ix = 1 , xTicMj + 1
0311                write(ChrDat,1) xrefmj(ix)/10.0**ilogx
0312                call GetStringWidth ( ChrDat , 255 , nchar , iwidth , ixchar )
0313                ipxref = hticmj(ix) - iwidth/2
0314                ipyref = ipymin - 3*FontData.ascent/2
0315                call MovAbs ( ipxref , ipyref )
0316                TxtOut = ChrDat(1:nchar)
0317                call DrawString ( %val(TxtOut) )
0318            end do
0319            call picComment ( %val(picGroupEnd) , %val(0) , %val(nil) )
0320
0321    c.....output reference values associated with y axis major tic marks
0322
0323            call picComment ( %val(picGroupBeg) , %val(0) , %val(nil) )
0324            do iy = 1 , yTicMj + 1
0325                call MovAbs ( ipxmin , vticmj(iy) )
0326                write(ChrDat,1) yrefmj(iy)/10.0**ilogy
0327                call GetStringWidth ( ChrDat , 255 , nchar , iwidth , ixchar )
0328                call MovRel (  iwidth , - FontData.ascent/2 )
0329                TxtOut = ChrDat(1:nchar)
0330                call DrawString ( %val(TxtOut) )
0331            end do
0332            call picComment ( %val(picGroupEnd) , %val(0) , %val(nil) )
0333
0334    c.....draw the flight path as well
0335
0336            if ( ntrp's.ge.1 ) then
0337                call DrawFlightPath ( xscale , xshift , yscale , yshift )
0338            end if
0339
0340    c.....draw the picture frame and lock the picture handle
0341
0342            call Filler
0343            call SetSize
0344            call HLock ( %val(PicHdl )
0345
0346    c.....terminate the update process
```

```
0347
0348            call EndUpdate ( %val(MapWPtr) )
0349
0350        1 format ( f6.1 )
0351
0352            return
0353            end


0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c----------------------------------------------------------------------------
0010            subroutine DrwAbs ( ix , iy )
0011    c----------------------------------------------------------------------------
0012    c     Draw to absolute graphics window position .
0013
0014            include 'PntAbs.inc'
0015            include 'WinLim.inc'
0016
0017            ixabs   = ix
0018            iyabs   = iy
0019
0020            call LineTo ( %val(ixabs) , %val(iGyMax-iyabs) )
0021
0022            return
0023            end


0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c----------------------------------------------------------------------------
0010            subroutine DshAbs ( ix , iy , icurve )
0011    c----------------------------------------------------------------------------
0012    c     Draw a dashed line to the point (ipx,ipy) from current pen location,
0013    c     clipping the portion outside the Map window.
0014
0015            include 'CrvDat.inc'
0016            include 'PntAbs.inc'
0017            include 'WinLim.inc'
0018
0019    c.....return if pen location will not change
0020
0021            if ( ix.eq.ixabs .and. iy.eq.iyabs ) then
0022                return
0023            end if
0024
0025    c.....initialize the end points of the segment to draw
0026
0027            ixold   = ixabs
0028            iyold   = iyabs
0029            ixnew   = ix
```

```
0030          iynew  = iy
0031
0032          xold   = float ( ixold )
0033          yold   = float ( iyold )
0034          xnew   = float ( ixnew )
0035          ynew   = float ( iynew )
0036
0037   c.....see if old pen location is inside the Map window
0038
0039          if ( ipxmin.le.ixold .and. ipxmax.ge.ixold .and.
0040         .      ipymin.le.iyold .and. ipymax.ge.iyold ) then
0041            locold = 0
0042          else
0043            locold = 1
0044          end if
0045
0046   c.....see if current pen location is inside the Map window
0047
0048          if ( ipxmin.le.ixnew .and. ipxmax.ge.ixnew .and.
0049         .      ipymin.le.iynew .and. ipymax.ge.iynew ) then
0050            locnew = 0
0051          else
0052            locnew = 1
0053          end if
0054
0055   c.....current pen location is inside window, new pen location is outside window.
0056   c      determine where line segment intersects window boundary.
0057
0058          if ( locold.eq.0 .and. locnew.eq.1 ) then
0059
0060   c........horizontal line intersects either left or right boundary of window
0061
0062             if ( iyold.eq.iynew ) then
0063               locnew = 0
0064               ixnew  = max0 ( ipxmin , min0 ( ipxmax , ixnew ) )
0065             end if
0066
0067   c........vertical line intersects either lower or upper boundary of window
0068
0069             if ( ixold.eq.ixnew ) then
0070               locnew = 0
0071               iynew  = max0 ( ipymin , min0 ( ipymax , iynew ) )
0072             end if
0073
0074   c........sloped line can intersect any of the four boundaries
0075
0076             if ( ixold.ne.ixnew .and. iyold.ne.iynew ) then
0077
0078   c...........determine slope and x intercept
0079
0080                slope  = ( ynew - yold )/( xnew - xold )
0081                b      = ynew - xnew*slope
0082
0083   c...........see if left boundary is intersected
0084
0085                if ( ixnew.lt.ipxmin ) then
0086                   iyhit  = inint ( b + slope*float(ipxmin) )
0087                   if ( iyhit.ge.ipymin .and. iyhit.le.ipymax ) then
0088                      locnew = 0
0089                      ixnew  = ipxmin
0090                      iynew  = iyhit
0091                   end if
0092                end if
0093
```

```
0094     c...........see if right boundary is intersected
0095
0096            if ( ixnew.gt.ipxmax ) then
0097                iyhit  = inint ( b + slope*float(ipxmax) )
0098                if ( iyhit.ge.ipymin .and. iyhit.le.ipymax ) then
0099                    locnew = 0
0100                    ixnew  = ipxmax
0101                    iynew  = iyhit
0102                end if
0103            end if
0104
0105     c...........see if lower boundary is intersected
0106
0107            if ( iynew.lt.ipymin ) then
0108                ixhit  = inint ( ( float(ipymin) - b )/slope )
0109                if ( ixhit.ge.ipxmin .and. ixhit.le.ipxmax ) then
0110                    locnew = 0
0111                    ixnew  = ixhit
0112                    iynew  = ipymin
0113                end if
0114            end if
0115
0116     c...........see if upper boundary is intersected
0117
0118            if ( iynew.gt.ipymax ) then
0119                ixhit  = inint ( ( float(ipymax) - b )/slope )
0120                if ( ixhit.ge.ipxmin .and. ixhit.le.ipxmax ) then
0121                    locnew = 0
0122                    ixnew  = ixhit
0123                    iynew  = ipymax
0124                end if
0125            end if
0126
0127        end if
0128     end if
0129
0130  c.....current pen location is outside window, new pen location is inside window.
0131  c     determine where line segment intersects window boundary.
0132
0133      if ( locold.eq.1 .and. locnew.eq.0 ) then
0134
0135  c........horizontal line intersects either left or right boundary of window
0136
0137            if ( iyold.eq.iynew ) then
0138                locold = 0
0139                ixold  = max0 ( ipxmin , min0 ( ipxmax , ixold ) )
0140            end if
0141
0142  c........vertical line intersects either lower or upper boundary of window
0143
0144            if ( ixold.eq.ixnew ) then
0145                locold = 0
0146                iyold  = max0 ( ipymin , min0 ( ipymax , iyold ) )
0147            end if
0148
0149  c........sloped line can intersect any of the four boundaries
0150
0151            if ( ixold.ne.ixnew .and. iyold.ne.iynew ) then
0152
0153  c...........determine slope and n intercept
0154
0155                slope  = ( ynew - yold ) / ( xnew - xold )
0156                b      = ynew - xnew*slope
0157
```

```
0158    c...........see if left boundary is intersected
0159
0160              if ( ixold.lt.ipxmin ) then
0161                  iyhit  = inint ( b + slope*float(ipxmin) )
0162                  if ( iyhit.ge.ipymin .and. iyhit.le.ipymax ) then
0163                     locold = 0
0164                     ixold  = ipxmin
0165                     iyold  = iyhit
0166                  end if
0167              end if
0168
0169    c...........see if right boundary is intersected
0170
0171              if ( ixold.gt.ipxmax ) then
0172                  iyhit  = inint ( b + slope*float(ipxmax) )
0173                  if ( iyhit.ge.ipymin .and. iyhit.le.ipymax ) then
0174                     locold = 0
0175                     ixold  = ipxmax
0176                     iyold  = iyhit
0177                  end if
0178              end if
0179
0180    c...........see if lower boundary is intersected
0181
0182              if ( iyold.lt.ipymin ) then
0183                  ixhit  = inint ( ( float(ipymin) - b )/slope )
0184                  if ( ixhit.ge.ipxmin .and. ixhit.le.ipxmax ) then
0185                     locold = 0
0186                     ixold  = ixhit
0187                     iyold  = ipymin
0188                  end if
0189              end if
0190
0191    c...........see if upper boundary is intersected
0192
0193              if ( iyold.gt.ipymax ) then
0194                  ixhit  = inint ( ( float(ipymax) - b )/slope )
0195                  if ( ixhit.ge.ipxmin .and. ixhit.ge.ipxmax ) then
0196                     locold = 0
0197                     ixold  = ixhit
0198                     iyold  = ipymax
0199                  end if
0200              end if
0201
0202           end if
0203        end if
0204
0205    c.....both end points are outside the window.  Determine whether line segment
0206    c       intersects window boundaries.
0207
0208        if ( locold.eq.1 .and. locnew.eq.1 ) then
0209
0210    c.........see if horizontal line intersects either left or right boundary of
0211    c         window
0212
0213              if ( iyold.eq.iynew ) then
0214                  if ( iyold.gt.ipymin .and. iyold.lt.ipymax ) then
0215                     locold = 0
0216                     locnew = 0
0217                     if ( ixold.le.ipxmin ) then
0218                        ixold  = ipxmin
0219                        ixnew  = ipxmax
0220                     else if ( ixold.ge.ipxmax ) then
0221                        ixold  = ipxmax
```

```
0222                          ixnew  = ipxmin
0223                       end if
0224                    end if
0225                 end if
0226
0227     c........see if vertical line intersects either lower or upper boundary of
0228     c        window
0229
0230              if ( ixold.eq.ixnew ) then
0231                 if ( ixold.gt.ipxmin  .and. ixold.lt.ipxmax ) then
0232                    locold = 0
0233                    locnew = 0
0234                    if ( iyold.le.ipymin ) then
0235                       iyold  = ipymin
0236                       iynew  = ipymax
0237                    else if ( iyold.ge.ipymax ) then
0238                       iyold  = ipymax
0239                       iynew  = ipymin
0240                    end if
0241                 end if
0242              end if
0243
0244     c........sloped line can intersect any of the four boundaries
0245
0246              if ( ixold.ne.ixnew .and. iyold.ne.iynew ) then
0247
0248     c..........determine slope and x intercept
0249
0250                 slope  = ( ynew - yold )/( xnew - xold )
0251                 b      = ynew - xnew*slope
0252
0253     c..........see if left boundary is intersected
0254
0255                 iyhit  = inint ( b + slope*float(ipxmin) )
0256                 if ( iyhit.ge.ipymin .and. iyhit.le.ipymax ) then
0257                    if ( ixold.lt.ixnew ) then
0258                       locold = 0
0259                       ixold  = ipxmin
0260                       iyold  = iyhit
0261                    else if ( ixold.gt.ixnew ) then
0262                       locnew = 0
0263                       ixnew  = ipxmin
0264                       iynew  = iyhit
0265                    end if
0266                 end if
0267
0268     c..........see if right boundary is intersected
0269
0270                 iyhit  = inint ( b + slope*float(ipxmax) )
0271                 if ( iyhit.ge.ipymin .and. iyhit.le.ipymax ) then
0272                    if ( ixold.lt.ixnew ) then
0273                       locnew = 0
0274                       ixnew  = ipxmax
0275                       iynew  = iyhit
0276                    else if ( ixold.gt.ixnew ) then
0277                       locold = 0
0278                       ixold  = ipxmax
0279                       iyold  = iyhit
0280                    end if
0281                 end if
0282
0283     c..........see if lower boundary is intersected
0284
0285                 ixhit  = inint ( ( float(ipymin) - b )/slope )
```

```
0286                    if ( ixhit.ge.ipxmin .and. ixhit.le.ipxmax ) then
0287                       if ( iyold.lt.iynew ) then
0288                          locold = 0
0289                          ixold  = ixhit
0290                          iyold  = ipymin
0291                       else if ( iyold.gt.iynew ) then
0292                          locnew = 0
0293                          ixnew  = ixhit
0294                          iynew  = ipymin
0295                       end if
0296                    end if
0297
0298  c...........see if upper boundary is intersected
0299
0300                    ixhit  = inint ( ( float(ipymax) - b )/slope )
0301                    if ( ixhit.ge.ipxmin .and. ixhit.le.ipxmax ) then
0302                       if ( iyold.lt.iynew ) then
0303                          locnew = 0
0304                          ixnew  = ixhit
0305                          iynew  = ipymax
0306                       else if ( iyold.gt.iynew ) then
0307                          locold = 0
0308                          ixold  = ixhit
0309                          iyold  = ipymax
0310                       end if
0311                    end if
0312
0313                 end if
0314              end if
0315
0316  c.....set the rectangle limits containing the line segment
0317
0318           ilxmin = min0 ( ix , ixabs )
0319           ilxmax = max0 ( ix , ixabs )
0320           ilymin = min0 ( iy , iyabs )
0321           ilymax = max0 ( iy , iyabs )
0322
0323  c.....draw visible part of dashed line segment
0324
0325           if ( locold.eq.0 .and. locnew.eq.0 ) then
0326
0327              if ( ixold.ge.ilxmin .and. ixold.le.ilxmax .and.
0328         .         iyold.ge.ilymin .and. iyold.le.ilymax .ard.
0329         .         ixnew.ge.ilxmin .and. ixnew.le.ilxmax .and.
0330         .         iynew.ge.ilymin .and. iynew.le.ilymax ) then
0331
0332                 call DashIt ( ixold , iyold , ixnew , iynew , icurve )
0333
0334              end if
0335
0336           end if
0337
0338  c.....save the current unclipped pen location
0339
0340           ixabs  = ix
0341           iyabs  = iy
0342
0343           return
0344           end


0001  c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003     !!G toolbox2.finc
```

```
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c----------------------------------------------------------------------
0010         subroutine EventHandler ( TheEvent , Option )
0011    c----------------------------------------------------------------------
0012    c    This routines figures out what kind of event has occurred and
0013    c    calls the appropriate routine to take action in response to the event.
0014
0015    c.....user option flags
0016
0017         include 'OptFlg.inc'
0018
0019         record / EventRecord /     TheEvent
0020         record / WindowPtr /       Wptr
0021
0022         integer*2                  Option
0023         integer*2                  WindowPart
0024
0025    c---------------------------------------------------------------------
0026
0027    c.....set option flag to continue
0028
0029         Option = oCycle
0030
0031    c.....event is MouseDown in menu region
0032
0033         if ( TheEvent.what.eq.mouseDown ) then
0034            WindowPart = FindWindow ( %val(TheEvent.where) , Wptr )
0035            if ( WindowPart.eq.inMenuBar ) then
0036               call DoMapMenu ( TheEvent , Option )
0037            end if
0038
0039    c.....key press events
0040
0041         else if ( TheEvent.what.eq.keyDown ) then
0042            call DoMapMenu ( TheEvent , Option )
0043
0044    c.....ignore all other events
0045
0046         else
0047         end if
0048
0049         return
0050         end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Main segment
0010
0011    !!S Main
0012
0013    c---------------------------------------------------------------------
Segment Main
0014         subroutine eventloop
```

79

```
0015    c-------------------------------------------------------------------------
0016    c     get events fo.. >ver and handle them.
0017
0018    !!SETC USINGINCLUDES = FALSE
0019          implicit none
0020
0021    c.....common block definition files
0022
0023          include 'AppleMenu.inc'
0024          include 'FileMenu.inc'
0025          include 'EditMenu.inc'
0026          include 'MapMenu.inc'
0027          include 'MBar.inc'
0028          include 'Globals.inc'
0029
0030    c.....define 2-byte zero for argument to FlushEvents toolbox routine
0031
0032          integer*2 zero /0/
0033
0034    c.....character items
0035
0036          integer*2              ChCode
0037          character*1            Ch
0038
0039    c.....menu pieces (menuResult = merge of menuID and menuItem)
0040
0041          integer*4 menuResult
0042          integer*4 itemMask /32767/
0043          integer*4 menuMask /-16/
0044          integer*2 menuID
0045          integer*2 menuItem
0046
0047    c.....code for event type
0048
0049          integer*2 code
0050
0051    c.....boolean for getting events
0052
0053          logical*1 gotEvent
0054
0055    c.....structures for window, event
0056
0057          record / WindowPtr   /      whichWindow
0058          record / EventRecord /      theEvent
0059
0060    c-------------------------------------------------------------------------
0061
0062    c.....we will start fresh on events
0063
0064          call FlushEvents ( %val(everyEvent), %val(zero) )
0065
0066    c.....enable the menus for "New Mission", "Open Mission"
0067
0068          call MenuSet ( FileMenuID, FileItemNewMission,  .true. )
0069          call MenuSet ( FileMenuID, FileItemOpenMission. .true. )
0070
0071    c.....set the flag for exiting program; we're not done yet
0072
0073          doneFlag = .false.
0074
0075    c.....initialize to no active text edit selection
0076
0077          theInput.TEH = nil
0078
```

```
0079    c.....************************
0080    c.....*** Main Event Loop ***
0081    c.....************************
0082
0083          do while (.true.)
0084
0085    c.....see if a TE is active
0086
0087          if ( theInput.TEH .ne. nil ) call TEIdle( %ref(theInput) )
0088
0089    c.....get a MultiFinder event
0090
0091          if (gHasWaitNextEvent) then
0092            gotEvent = WaitNextEvent( %val(everyEvent), %ref(theEvent),
0093          &                          int4(SleepValue), %val(nil)      )
0094          else
0095            call SystemTask
0096            gotEvent = GetNextEvent( %val(everyEvent), %ref(theEvent))
0097          end if
0098
0099    c.....work the event
0100
0101          if (gotEvent) then
0102            select case (theEvent.what)
0103
0104              case (mouseDown)
0105
0106    c..........find the window in which the event happened
0107                code = FindWindow( %val(theEvent.where), %ref(whichWindow) )
0108
0109                select case (code)
0110                  case (inMenuBar)
0111
0112    c...............determine the menu selection
0113                    menuResult = MenuSelect ( %ref(theEvent.where) )
0114
0115    c...............extract the menu and item numbers from within menuResult
0116                    menuItem = jiand ( menuResult, itemMask )
0117                    menuID   = jishft( menuResult, menuMask )
0118    c...............call my Menu handler
0119                    call DoMenu (menuID, menuItem)
0120
0121                  case (inDrag)
0122    cc                call DoDrag(whichWindow)
0123
0124                  case (inGrow)
0125    cc                call DoGrow(whichWindow)
0126
0127                  case (inGoAway)
0128    cc                call DoGoAway(whichWindow)
0129
0130                  case (inContent)
0131    cc                call DoInContent(whichWindow)
0132
0133                  case (inSysWindow)
0134                    call SystemClick( %ref(theEvent), %ref(whichWindow) )
0135
0136                end select !code
0137
0138              case (keyDown, autoKey)
0139                call DoKeyEvent (theEvent)
0140
0141              case (UpdateEvt)
0142    cc            call DoUpdate
```

```
0143
0144            case (DiskEvt)
0145   cc           call DoDiskEvent
0146
0147            case (ActivateEvt)
0148   cc           call DoActivate
0149
0150            case default
0151
0152         end select !theEvent.what
0153
0154   c.....end of if construct for gotEvent
0155
0156         endif
0157
0158   c.....******************************
0159   c.....*** end of Main Event Loop ***
0160   c.....******************************
0161
0162         end do
0163
0164         return
0165         end
```

```
0001   !!T72+
0002         SUBROUTINE GEN4D                                          GEN4   1
0003   C.....GENERATES NG = 9 OR 16 4D PROFILES P,D,T AND SIGMAS SP,SD,ST AT  GEN4   2
0004   C         GRID OF LATITUDES AND LONGITUDES GLAT,GLON. CURRENT LATITUDE,  GEN4   3
0005   C           LONGITUDE=CLAT,CLON. PREVIOUS LATITUDE,LONGITUDE=PLAT,PLON.  GEN4   4
0006         COMMON/C4/GLAT(16),GLON(16),NG,P(16,26),D(16,26),T(16,26),  GEN4   5
0007       $ SP(16,26),SD(16,26),ST(16,26),PLON,CLON,HS                 GEN4   6
0008         COMMON/IOTEMP/IOTEM1,IOTEM2 IUG,IUN,DDD,XMJD,PLAT,CLAT,     GEN4   7
0009       $ NSAME,RP1,RD1,RT1,SP1,SD1,ST1,RU1,RV1,SU1,SV1,             GEN4   8
0010       $ MN,IDA,IYR,H1,PHI1R,THET1R,GZ,RI,Z,PHIR,THETR,F10,F10B,AP, GEN4   9
0011       $ IHR,MIN,NMORE,DX,HL,VL,DZ,B,EPS,IOPP,LOOK,DUMMY(20)         GEN4  10
0012         COMMON/PDTCOM/IU4,MONTH,IOPR,PG(18,19),TG(18,19),DG(18,19), GEN4  11
0013       1      FSP(8,10,12),DSP(8,10,12),TSP(8,10,12)                GEN4  12
0014       2,PAQ(17,5),DAQ(17,5),TAQ(17,5),                            GEN4  13
0015       3PDQ(17,5),DDQ(17,5),TDQ(17,5),PR(20,10),DR(20,10),TR(20,10),GEN4  14
0016       4UAQ(17,5),VAQ(17,5),UDQ(17,5),VDQ(17,5),UR(25,10),VR(25,10),GEN4  15
0017       5PQ,DQ,TQ,UQ,VQ,PQA,DQA,TQA,UA,VA,IOPQ                      GEN4  16
0018       * ,PLP(25,10),DLP(25,10),TLP(25,10),ULP(25,10),VLP(25,10),UDL(25, GEN4  17
0019       * 10),VDL(25,10),UDS(25,10),VDS(25,10)                       GEN4  18
0020         COMMON/ADJCOM/DUM(130),KOUNT                               GEN4  19
0021         COMMON/IPRTP/ IPRT                                         GEN4  20
0022         DIMENSION NGOOD(26)                                        GEN4  20B
0023         IF(NSAME.EQ.1) RETURN                                      GEN4  21
0024         IPRT=0                                                     GEN4  22
0025         LOOK=0                                                     GEN4  23
0026         F = 0.017453293                                            GEN4  24
0027         NG = 16                                                    GEN4  25
0028         DX = PLON - CLON                                           GEN4  26
0029         IF(DX.GT.180.0) DX = DX - 360.0                            GEN4  26B
0030         IF(DX.LT.-180.0)DX = DX + 360.0                            GEN4  26C
0031   C.....LONGITUDE DISPLACEMENT FROM PREVIOUS TO CURRENT POSITION   GEN4  27
0032         DY = CLAT - PLAT                                           GEN4  28
0033   C.....LATITUDE DISPLACEMENT FROM PREVIOUS TO CURRENT POSITION    GEN4  29
0034         IF (DY) 20,10,20                                           GEN4  30
0035      10  IF (DX) 15,12,15                                          GEN4  31
0036      12  K = 0                                                     GEN4  32
0037         GO TO 40                                                   GEN4  33
0038      15 THETA = 180. + SIGN(90.,DX)                                GEN4  34
0039         GO TO 30                                                   GEN4  35
```

```
0040        20 THETA = ATAN(DX/DY)/F                                    GEN4  36
0041           IF (DY.GT.0.) THETA = THETA + 180.                       GEN4  37
0042           IF (THETA.LT.0.) THETA = THETA + 360.                    GEN4  38
0043   C.....THETA = AZIMUTH ANGLE OF TRAJECTORY, USED TO ORIENT LAT-LON GRID  GEN4  39
0044        30 K = INT((THETA + 67.5)/45.)                             GEN4  40
0045   C       INDEX USED IN COMPUTED GO TO FOR 110 THRU 180            GEN4  41
0046           IF (K.GT.8) K=K-8                                        GEN4  42
0047   C       NORTH POLAR GRID                                         GEN4  43
0048           IF (CLAT.GT.75.0.AND.K.GE.3.AND.K.LE.7)GO TO 200         GEN4  44
0049   C       SOUTH POLAR GRID                                         GEN4  45
0050           IF (CLAT.LT.-75.0.AND.(K.GE.7.OR.K.LE.3))GO TO 200       GEN4  46
0051   C.....INITIAL ESTIMATE OF REFERNCE LATITUDE (LOWER LEFT GRID POINT)  GEN4  47
0052    40     LAT0 = 5*INT(CLAT/5.)                                    GEN4  48
0053           IF (CLAT.LT.0.) LAT0 = LAT0 - 5                          GEN4  49
0054   C.....INITIAL ESTIMATE OF REFERENCE LONGITUDE (LOWER LEFT GRID POINT)  GEN4  50
0055           LON0=5*INT(CLON/5.)                                      GEN4  51
0056   C.....ADJUSTS LAT0,LON0 ACCORDING TO DIRECTION OF TRAJECTORY AZIMUTH  GEN4  52
0057           IF (K.GT.0) GO TO 100                                    GEN4  53
0058           LAT0 = LAT0 - 5                                          GEN4  54
0059           LON0= LON0 + 10                                          GEN4  55
0060           GO TO 190                                                GEN4  56
0061    100    GO TO (110,120,130,140,150,160,170,180),K                GEN4  57
0062    110 LAT0 = LAT0-10                                              GEN4  58
0063           LON0 = LON0 + 10                                         GEN4  59
0064           GO TO 190                                                GEN4  60
0065    120 LAT0 = LAT0-10                                              GEN4  61
0066           LON0 = LON0+15                                           GEN4  62
0067           GO TO 190                                                GEN4  63
0068    130 LAT0 = LAT0-5                                               GEN4  64
0069           LON0 = LON0+15                                           GEN4  65
0070           GO TO 190                                                GEN4  66
0071    140 LON0 = LON0+15                                              GEN4  67
0072           GO TO 190                                                GEN4  68
0073    150 LON0 = LON0+10                                              GEN4  69
0074           GO TO 190                                                GEN4  70
0075    160 LON0 = LON0+5                                               GEN4  71
0076           GO TO 190                                                GEN4  72
0077    170 LAT0 = LAT0-5                                               GEN4  73
0078           LON0 = LON0+5                                            GEN4  74
0079           GO TO 190                                                GEN4  75
0080    180 LAT0 = LAT0-10                                              GEN4  76
0081           LON0 = LON0+5                                            GEN4  77
0082    190 IF (LON0.GE.360) LON0 = LON0 - 360                         GEN4  79
0083           IF (LAT0.GT.75)  LAT0 = 75                               GEN4  78B
0084           DLI=1.25                                                 GEN4  79
0085           IF(ABS(CLAT).GE.18) GO TO 192                            GEN4  80
0086           DLI=3.0                                                  GEN4  81
0087           LAT0=-18                                                 GEN4  82
0088    192 DO 195 I=1,4                                                GEN4  83
0089           I12 = I+12                                               GEN4  84
0090           DO 195 J=I,I12,4                                         GEN4  85
0091           GLAT(J) = LAT0 + DLI*(J-I)                               GEN4  86
0092   C.....LATITUDE, LONGITUDE GRID AT 5 DEGREE INTERVALS             GEN4  87
0093    195 GLON(J) = LON0 - 5. * (I - 1)                               GEN4  88
0094           GO TO 400                                                GEN4  89
0095   C       POLAR GRID                                               GEN4  90
0096    200 NG = 9                                                      GEN4  91
0097           DO 210 J=1,8                                             GEN4  92
0098   C.....POLAR GRID LATITUDES 1-8 = +75 (N) OR -75 (N)              GEN4  93
0099           GLAT(J) = SIGN(75.,CLAT)                                 GEN4  94
0100   C.....POLAR GRID LONGITUDES 1-8 AT 45 DEG INTERVALS              GEN4  95
0101    210 GLON(J) = 45.*(J-1)                                         GEN4  96
0102   C.....POLAR GRID LATITUDE 9 = POLE +93 OR -90                    GEN4  97
0103           GLAT(9) = SIGN(90.,CLAT)                                 GEN4  98
```

```
0104    C.....POLAR GRID LONGITUDE 9 = 0                                        GEN4   99
0105          GLON(9) = 0.                                                      GEN4  100
0106    C.....GENERATES 16 PROFILES (OR 9 PROFILES FOR POLAR GRID)              GEN4  101
0107      400 CALL GRID4D                                                       GEN4  102
0108          DO 390 I = 1,NG                                                   GEN4102B
0109          DO 330 J = 1,26                                                   GEN4102C
0110          NGOOD(J) = 1                                                      GEN4102D
0111          IF(P(I,J).LE.0.0.OR.D(I,J).LE.0.0.OR.T(I,J).LE.0.0)NGOOD(J)=0     GEN4102E
0112          IF(NGOOD(J).EQ.0)GOTO 330                                         GEN4102F
0113          RATIO = P(I,J)/(D(I,J)*T(I,J))                                    GEN4102G
0114          IF(RATIO.GT.286.0.AND.RATIO.LT.288.0)GOTO 330                     GEN4102H
0115          NGOOD(J) = 0                                                      GEN4102I
0116          WRITE(6,325)I,J,RATIO                                             GEN4102J
0117      325 FORMAT(' GAS LAW VIOLATION. I,J,RATIO = ',2I4,G12.4)             GEN4102K
0118      330 CONTINUE                                                          GEN4102L
0119          DO 340 J = 3,26                                                   GEN4102M
0120          IF(NGOOD(J).EQ.0.OR.NGOOD(J-1).EQ.0)GOTO 340                      GEN4102N
0121          DENOM = 1./T(I,J)                                                 GEN4102O
0122          IF(ABS(T(I,J)-T(I,J-1)).GT.0.01)DENOM=ALOG(T(I,J-1)/T(I,J))/      GEN4102P
0123         &  (T(I,J-1)-T(I,J))                                               GEN4102Q
0124          RATIO = ALOG(P(I,J-1)/P(I,J))/DENOM                               GEN4102R
0125          IF(RATIO.GT.30.7.AND.RATIO.LT.37.6)GOTO 340                       GEN4102S
0126          NGOOD(J) = 0                                                      GEN4102T
0127          WRITE(6,335)I,J,RATIO                                             GEN4102U
0128      335 FORMAT(' HYDROSTATIC VIOLATION. I,J,RATIO = ',2I4,G12.4)         GEN4102V
0129          IF(J.EQ.26)GO TO 345                                             GEN4102W
0130          K1 = J + 1                                                        GEN4102X
0131          DO 336 K = K1,26                                                  GEN4102Y
0132      336 NGOOD(K) = 0                                                      GEN4102Z
0133          GO TO 345                                                         GEN4102a
0134      340 CONTINUE                                                          GEN4102b
0135      345 NBAD = 0                                                          GEN4102c
0136          DO 360 J = 1,26                                                   GEN4102d
0137          IF(NGOOD(J).GT.0)GOTO 360                                         GEN4102e
0138          NBAD = NBAD + 1                                                   GEN4102f
0139          P(I,J) = 0.                                                       GEN4102g
0140          D(I,J) = 0.                                                       GEN4102h
0141          T(I,J) = 0.                                                       GEN4102i
0142      360 CONTINUE                                                          GEN4102j
0143          IF(NBAD.LE.12)GOTO 390                                            GEN4102k
0144          WRITE(6,380)                                                      GEN4102l
0145      380 FORMAT(' UNABLE TO GENERATE 4-D GRID. TOO MANY TEST VIOLATIONS') GEN4102m
0146          STOP                                                              GEN4102n
0147      390 CONTINUE                                                          GEN4102o
0148          DO 600 I=1,NG                                                     GEN4  103
0149          IHV = 0                                                           GEN4  104
0150          SPR = 0.0004                                                      GEN4  105
0151          SDR = 0.0004                                                      GEN4  106
0152          STR = 0.0004                                                      GEN4  107
0153          DO 420 J = 8,26                                                   GEN4  108
0154          CHECK = 1.                                                        GEN4  109
0155          IF(P(I,J).LE.0.0.OR.SP(I,J).LE.0.0)CHECK = 0.                     GEN4  110
0156          IF(D(I,J).LE.0.0.OR.SD(I,J).LE.0.0)CHECK = 0.                     GEN4  111
0157          IF(T(I,J).LE.0.0.OR.ST(I,J).LE.0.0)CHECK = 0.                     GEN4  112
0158    C.... FINDS INDEX IHV OF LAST HEIGHT ABOVE 6 KM WITH NON-ZERO DATA      GEN4  113
0159          IF(CHECK.GT.0.)GO TO 420                                          GEN4  114
0160          IHV = J-1                                                         GEN4  115
0161          GO TO 440                                                         GEN4  116
0162      420 CONTINUE                                                          GEN4  117
0163    C     HEIGHT = HEIGHT INDEX - 1                                         GEN4  118
0164      440 Z1 = IHV -1.                                                      GEN4  119
0165          IF(IHV.EQ.0)GO TO 491                                            GEN4119B
0166    C     SPR,SDR,STR=SIGMAS AT HEIGHT Z1                                   GEN4  120
0167          SPR = SP(I,IHV)                                                   GEN4  121
```

```
0168             SDR=SD(I IHV)                                              GEN4 122
0169             STR=ST(I,IHV)                                             GEN4 123
0170             IF(SPR.LE.0.0)SPR = 0.0004                                GEN4123B
0171             IF(SDR.LE.0.0)SDR = 0.0004                                GEN4123C
0172             IF(STR.LE.0.0)STR = 0.0004                                GEN4123D
0173             IF(IHV.GT.12)GOTO 441                                     GEN4123E
0174             WRITE(6,442)IHV                                           GEN4123F
0175         442 FORMAT(' UNABLE TO GENERATE 4-D GRID. IHV = ',I3)         GEN4123G
0176             STOP                                                      GEN4123H
0177         441 CONTINUE                                                  GEN4123I
0178   C.....IF HEIGHT Z1 GEQ 20 KM, USE GROVES AT 30 KM FOR INTERPOLATION, GEN4 124
0179   C          OTHERWISE USE GROVES AT 25 KM                           GEN4 125
0180             IF (IHV.GE.21) GO TO 480                                  GEN4 126
0181   C.....EVALUATES GROVES AT 25 KM FOR INTERPOLATION AND              GEN4 127
0182   C          FILL IN OF ZERO DATA                                    GEN4 128
0183             CALL GTERP(25,GLAT(I),P2,D2,T2,PG,DG,TG,DPY,DTY,DP2Y)     GEN4 129
0184             IHP = IHV + 1                                            GEN4 130
0185             DO 450 K=IHP,26                                          GEN4 131
0186   C.....AVOIDS INTERPOLATION OF P,D,T IF ONLY SIGMAS ARE ZERO        GEN4 132
0187             IF(P(I,K).GT.0.0.AND.D(I,K).GT.0.0.AND.T(I,K).GT.0.0)GO TO 445 GEN4 133
0188             H=K-1                                                     GEN4 134
0189   C.....INTERPOLATES BETWEEN 4D AT HEIGHT Z1 AND GROVES AT 25 TO FILL GEN4 135
0190   C          IN MISSING DATA                                         GEN4 136
0191             CALL INTEP2(P(I,IHV),D(I,IHV),T(I,IHV),Z1,P2,D2,T2,25.,PH,DH,TH,H)GEN4 137
0192             P(I,K)=PH                                                 GEN4 138
0193             D(I,K)=DH                                                 GEN4 139
0194             T(I,K)=TH                                                 GEN4 140
0195         445 SP(I,K) = SPR                                            GEN4 141
0196             SD(I,K)=SDR                                               GEN4 142
0197   C.....SETS MISSING SIGMAS EQUAL TO SIGMAS AT HEIGHT Z1             GEN4 143
0198         450 ST(I,K)=STR                                             GEN4 144
0199             GO TO 491                                                GEN4 145
0200   C.....EVALUATES GROVES AT 30 KM FOR INTERPOLATION AND FILL IN OF   GEN4 146
0201   C          ZERO DATA                                               GEN4 147
0202         480 CALL GTERP(30,GLAT(I),P2,D2,T2,PG,DG,TG,DPY,DTY,DP2Y)     GEN4 148
0203   C        COMPUTE PERTURBATIONS TO GROVES MODEL                     GEN 148B
0204             CALL PDTUV(PSP,DSP,TSP,GLAT(I),GLON(I),30,DP,DD,DT,DFX,DPY,DTX,DTYGEN4 149
0205           $ ,DP2X,DP2Y,DPXY)                                         GEN4 151
0206   C.....ADD STATIONARY PERTURBATIONS TO GROVES MODEL                 GEN4 152
0207             P2 = P2*(1. + DP)                                        GEN4 153
0208             D2 = D2*(1. + DD)                                        GEN4 154
0209             T2 = T2*(1. + DT)                                        GEN4 155
0210             IHP = IHV + 1                                            GEN4 156
0211             DO 490 K=IHP,26                                          GEN4 157
0212   C.....AVOIDS INTERPOLATING P,D,T IF ONLY SIGMAS ARE ZERO           GEN4 158
0213             IF(P(I,K).GT.0.0.AND.D(I,K).GT.0.0.AND.T(I,K).GT.0.0)GO TO 485 GEN4 159
0214             H=K-1                                                     GEN4 160
0215   C.....INTERPOLATES BETWEEN 4D AT HEIGHT Z1 AND GROVES AT 30 KM TO  GEN4 161
0216   C          FILL IN MISSING DATA                                    GEN4 162
0217             CALL INTEP2(P(I,IHV),D(I,IHV),T(I,IHV),Z1,P2,D2,T2,30.,PH,DH,TH,H)GEN4 163
0218             P(I,K)=PH                                                 GEN4 164
0219             D(I,K)=DH                                                 GEN4 165
0220             T(I,K)=TH                                                 GEN4 166
0221         485 SP(I,K) = SPR                                            GEN4 167
0222             SD(I,K)=SDR                                               GEN4 168
0223   C        SET MISSING SIGMAS AT HEIGHT 1                            GEN4 169
0224         490 ST(I,K) = STR                                           GEN4 170
0225         491 CONTINUE                                                  GEN4 171
0226             IHP = IHV - 1                                            GEN4 172
0227             SPO = SP(I,1)                                            GEN4 173
0228             SDO = SD(I,1)                                            GEN4 174
0229             STO = ST(I,1)                                            GEN4 175
0230             IF(SPO.LE.0.0)SPO = 0.0001                                GEN4 176
0231             IF(SDO.LE.0.0)SDO = 0.0001                                GEN4176B
```

85

```
0232          IF(ST0.LE.0.0)ST0 = 0.0001                          GEN4176C
0233          DO 492 K = 1,9                                       GEN4176D
0234          IF(SP(I,K) .LE. 0.) SP(I,K) = SP0                   GEN4176E
0235          IF(SD(I,K) .LE. 0.) SD(I,K) = SD0                   GEN4176F
0236      492 IF(ST(I,K) .LE. 0.) ST(I,K) = ST0                   GEN4176G
0237          DO 495 K=10,IHP                                      GEN4 177
0238    C......SETS ALL ZERO SIGMAS TO SIGMA AT HEIGHT Z1          GEN4 178
0239          IF (SP(I,K).LE.0.0.AND.P(I,K).GT.0.) SP(I,K) = SPR  GEN4 179
0240          IF (SD(I,K).LE.0.0.AND.D(I,K).GT.0.) SD(I,K) = SDR  GEN4 180
0241      495 IF (ST(I,K).LE.0.0.AND.T(I,K).GT.0.) ST(I,K) = STR  GEN4 181
0242      500 PA = P(I,1)                                          GEN4 182
0243          TA = T(I,1)                                          GEN4 183
0244          R =287.05                                            GEN4 184
0245          G = GZ*(1.+(Z/(RI-Z)))**2                            GEN4 185
0246          K = 2                                                GEN4 186
0247      510 PB = P(I,K)                                          GEN4 187
0248          TB = T(I,K)                                          GEN4 188
0249          IF ((PB*TB) .GT. 0.) GO TO 520                       GEN4 189
0250          K = K + 1                                            GEN4 190
0251          GO TO 510                                            GEN4 191
0252      520 IF(ABS(TA-TB).LE.0.01)GOTO 570                       GEN4 192
0253      560 IF(TA*TB.LE.0.0)GO TO 570                            GEN4 193
0254          TZ = (TA-TB) / ALOG(TA/TB)                           GEN4193B
0255          GO TO 575                                            GEN4 194
0256      570 TZ = TA                                              GEN4 195
0257      575 HS = K - 1.                                          GEN4 196
0258          IF(PB*PA.LE.0.0)GO TO 576                            GEN4 197
0259          HS = K - 1. + 0.001*R*TZ*ALOG(PB/PA)/G               GEN4197B
0260      576 KM = K - 2                                           GEN4197C
0261          IF(ABS(K-1-HS).GT.0.1) GO TO 578                     GEN4 198
0262          GAM=TB-T(I,K+1)                                      GEN4 199
0263          IF(ABS(GAM).LE.0.01)GOTO 590                         GEN4 200
0264          GO TO 582                                            GEN4 201
0265      578 IF(ABS(TA-TB).LE.0.01)GOTO 590                       GEN4201B
0266      580 GAM=(TA-TB)/(K-1-HS)                                 GEN4 202
0267      582 KM1=KM+1                                             GEN4 203
0268          IF(ABS(GAM).GT.G) GAM=SIGN(G,GAM)                    GEN4 204
0269          DO 585 JD=1,KM1,1                                    GEN4 205
0270          J=JD-1                                               GEN4 206
0271          TJ=TA-GAM*(J-HS)                                     GEN4 207
0272          PJ=PA*(TJ/TA)**(G/(R*GAM*0.001))                     GEN4 208
0273          DJ=PJ/(R*TJ)                                         GEN4 209
0274          P(I,J+1)=PJ                                          GEN4 210
0275          D(I,J+1)=DJ                                          GEN4 211
0276      585 T(I,J+1)=TJ                                          GEN4 212
0277          GO TO 599                                            GEN4 213
0278      590 KM1=KM+1                                             GEN4 214
0279          DO 595 JD=1,KM1,1                                    GEN4 215
0280          J=JD-1                                               GEN4 216
0281          TJ=TA                                                GEN4 217
0282          PJ=PA*EXP(-G*(J-HS)/(R*0.001*TJ))                    GEN4 218
0283          DJ=PJ/(R*TJ)                                         GEN4 219
0284          P(I,J+1)=PJ                                          GEN4 220
0285          D(I,J+1)=DJ                                          GEN4 221
0286      595 T(I,J+1)=TJ                                          GEN4 222
0287          IF(NSAME.EQ.2) NSAME=1                               GEN4 223
0288      599 HS=0.                                                GEN4 224
0289          KOUNT = I                                            GEN4 225
0290          CALL ADJUST                                          GEN4 226
0291      600 CONTINUE                                             GEN4 227
0292          RETURN                                               GEN4 228
0293          END                                                 GEN4 229
0294          SUBROUTINE GRID4D                                    GRID    1
0295          REAL LAT,LON                                         GRID    3
```

```
0296            COMMON/C4/LAT(16),LON(16),NP,P(16,26),R(16,26),T(16,26),SP(16,26),GRID   4
0297          $ SR(16,26),ST(16,26),DU1,DU2,DUMMY                               GRID   5
0298            COMMON /PDTCOM/ IT,MONTH,DUMMY1(8118)                           GRID   6
0299    C                                                                       GRID   7
0300    C                                                                       GRID   8
0301    C       SUBROUTINE TO SELECT PRESSURE, TEMPERATURE, AND DENSITY PROFILES (GRID 9
0302    C       TOGETHER WITH THE NORMALIZED VARIANCES IN EACH, AT UP TO 16  GRID GRID  10
0303    C       AT LAT/LONS SELECTED BY CALLING PROGRAM.                       GRID  11
0304    C                                                                       GRID  12
0305    C       USES NASA HUNTSVILLE MSFC 4-D DATA FILES                       GRID  13
0306    C                                                                       GRID  14
0307            DIMENSION IN(215)
0308    C                                                                       GRID  16
0309            COMMON /IOTEMP/ IOTEM1,IOTEM2,DUMMY2(62)                       GRID  17
0310            COMMON /POINT/ IPT(16,5),LL(16),DXY(16,2)                      GRID  18
0311            COMMON /ORDER/ IPTN(16,5),IREAD(65,3)                          GRID  19
0312            COMMON /INT/ D(208,5),IG(5),DYX(2),DLA(4),DLO(4)               GRID  20
0313    C                                                                       GRID  21
0314            INTEGER IOTEM1                                                  GRID  22
0315            CHARACTER*6  M
0316            CHARACTER*2  CMONTH
0317    C                                                                       GRID  23
0318            WRITE (CMONTH,'(I2)') MONTH
0319            IF(MONTH.LT.10)THEN
0320               OPEN(UNIT=IT,FILE='M'
0321          1         //CMONTH(2:2)//'.DAT',RECL=213,STATUS='OLD',
0322          2         FORM='UNFORMATTED')
0323               ELSE
0324               OPEN(UNIT=IT,FILE='M'
0325          1         //CMONTH//'.DAT',RECL=213,STATUS='OLD',
0326          2         FORM='UNFORMATTED')
0327               ENDIF
0328    C                                                                       GRID  24
0329    C       INITIALIZE                                                      GRID  25
0330    C                                                                       GRID  26
0331            ZERO=0.0                                                        GRID  27
0332            ONE=1.0                                                         GRID  28
0333            TEN=10.0                                                        GRID  29
0334            HUNDR=100.0                                                     GRID  30
0335            THOU=1000.0                                                     GRID  31
0336    C                                                                       GRID  34
0337            N=MONTH-1-((2*MONTH)/9)*4                                       GRID  35
0338            IF(MONTH.EQ.13) N=0                                             GRID  36
0339            NUMEOF = 0                                                      GRID  37
0340    C                                                                       GRID  41
0341    C       CAUTION :   APPROPRIATE 4-D INPUT DATA FILE MUST BE ASSIGNED TO  GRID  42
0342    C                   PROPER UNIT BEFORE READING IN DATA.                 GRID  43
0343    C                                                                       GRID  44
0344            REWIND (IT)
0345    C
0346        20 CALL SELEC4                                                      GRID  45
0347    C                                                                       GRID  46
0348            IPC=0                                                           GRID  47
0349            IRN=1                                                           GRID  48
0350            IF(IREAD(IRN,3).EQ.0) GO TO 39                                  GRID  49
0351        21 JT=IT                                                            GRID  50
0352            M=' READ '                                                      GRID  51
0353        22 READ(IT,ERR=50,END=39) (IN(I),I=1,213)                          GRID  52
0354            IPC =IPC +1                                                     GRID  53
0355            GO TO 60
0356        50 WRITE(6,23) IT, IPC                                             GRID  55
0357        23 FORMAT(1H ,' INPUT UNIT NO.',I3,' IN ERROR FOR RECORD NO.  ',I5)
0358        60 CONTINUE
0359            IF(IRN.GT.IREAL(IRN,3)) GO TO 22                               GRID  58
```

```
0360            IF(IRC.GT.IREAD(IRN,3)) GO TO 39              GRID  59
0361       24 I=IREAD(IRN,1)                                 GRID  60
0362          J=IREAD(IRN,2)                                 GRID  61
0363          IF(IRN.EQ.1) GO TO 25                          GRID  62
0364          IF(IREAD(IRN,3).EQ.IREAD(IRN-1,3)) GO TO 27    GRID  63
0365       25 IP=IN(212)                                     GRID  64
0366          MP=IN(213)                                     GRID  65
0367          IF((MP.NE.MONTH).OR.(IP.NE.IPT(I,J))) GO TO 39 GRID  66
0368          DO 26 K=213,1,-1                               GRID  67
0369          IN(K+2)=IN(K)                                  GRID  69
0370       26 CONTINUE
0371       27 IN(1) = I
0372          IN(2) = J                                      GRID  72
0373          JT=IOTEM1                                      GRID  73
0374          M=' WRITE'                                     GRID  74
0375          WRITE(IOTEM1) IN                               GRID  75
0376          IRN=IRN+1                                      GRID  76
0377          IF(IREAD(IRN,3).EQ.IRC) GO TO 24               GRID  77
0378          IF(IREAD(IRN,3).EQ.0) GO TO 28                 GRID  78
0379          GO TO 21                                       GRID  79
0380    C                                                    GRID  80
0381    C     INTERPOLATE TO GIVEN LAT/LON FROM GRID DATA    GRID  81
0382    C                                                    GRID  82
0383       28 M=' READ '                                     GRID  83
0384          DO 38 II=1,NP                                  GRID  84
0385          DO 29 I=1,208                                  GRID  85
0386          DO 29 J=1,5                                    GRID  86
0387          D(I,J)=0.0                                     GRID  87
0388       29 CONTINUE                                       GRID  88
0389          DO 32 J=1,4                                    GRID  89
0390          IF(IPT(II,J).EQ.0) GO TO 32                    GRID  90
0391          INDEX1 = II
0392          INDEX2 = J
0393          REWIND (IOTEM1)                                GRID  93
0394       30 READ(IOTEM1,END=39) IN                         GRID  94
0395          IF(IN(1) .NE. INDEX1  .OR.  IN(2) .NE. INDEX2) GO TO 30  GRID  95
0396          DO 31 I=3,210                                  GRID  96
0397          D(I-2,J) = IN(I)/HUNDR                         GRID  99
0398       31 CONTINUE                                       GRID 101
0399          DLA(J) = IN(211)/TEN                           GRID 102
0400          DLO(J) = IN(212)/TEN                           GRID 103
0401       32 CONTINUE                                       GRID 104
0402    C                                                    GRID 105
0403    C     IF NECESSARY, INTERPOLATE                      GRID 106
0404    C                                                    GRID 107
0405          LALO=LL(II)                                    GRID 108
0406          DO 33 I=1,5                                    GRID 109
0407          IG(I)=IPT(II,I)                                GRID 110
0408       33 CONTINUE                                       GRID 111
0409          IF(IG(2).NE.0) GO TO 35                        GRID 112
0410          DO 34 I=1,208                                  GRID 113
0411          D(I,5)=D(I,1)                                  GRID 114
0412       34 CONTINUE                                       GRID 115
0413          GO TO 37                                       GRID 116
0414       35 IF(IG(5).NE.2) GO TO 36                        GRID 117
0415          DYX(1)=DXY(II,1)                               GRID 118
0416          DYX(2)=DXY(II,2)                               GRID 119
0417    C                                                    GRID 120
0418       36 CALL INTRP4 (LALO)                             GRID 121
0419    C                                                    GRID 122
0420       37 DO 38 I=1,26                                   GRID 123
0421          P(II,I)=D(I,5)*HUNDR                           GRID 124
0422          P(II,I)=D(I+156,5)*HUNDR                       GRID 125
0423          T(II,I) =D(I+52,5)                             GRID 126
```

88

```
0424          DIVIDE=ONE                                                  GRID 127
0425          IF(P(II,I).GT.ZERO) DIVIDE=(P(II,I)/HUNDR)**2               GRID 128
0426          SP(II,I)=D(I+26,5)/DIVIDE                                   GRID 129
0427          DIVIDE=ONE                                                  GRID 130
0428          IF(R(II,I).GT.ZERO) DIVIDE=(THOU*R(II,I))**2               GRID 131
0429          SR(II,I)=D(I+182,5)/DIVIDE                                  GRID 132
0430          DIVIDE=ONE                                                  GRID 133
0431          IF(T(II,I).GT.ZERO) DIVIDE=T(II,I)**2                       GRID 134
0432          ST(II,I)=D(I+78,5)/DIVIDE                                   GRID 135
0433       38 CONTINUE                                                    GRID 136
0434          REWIND (IOTEM1)                                            GRID136B
0435          RETURN                                                      GRID 137
0436       39 I = IREAD(IRN,1)                                          GRID137B
0437          J = IREAD(IRN,2)                                          GRID137C
0438          WRITE(6,40) JT,IRC,IREAD(IRN,3),MP,MONTH,IP,I,J,IPT(I,J),IRN,M  GRID 138
0439       40 FORMAT(1X,'***** UNIT NO. ',I3,' IN ERROR ',I7,' RECORDS READ'/
0440        1 'IREAD(IRN,3) =',I5,' MP = ',I3,' MONTH = ',I3,' IP = ',I5,
0441        2 ' IPT(',I2,',',I1,') = ',I5,' IRN =',I3/' STATUS: ',A6)
0442          STOP 'EXECUTION TERMINATED DUE TO ERROR CONDITION'          GRID 142
0443          END                                                        GRID 143
0444          SUBROUTINE ADJUST                                          ADJU   1
0445          COMMON/C4/DUN1(32),NG,P(16,26),D(16,26),T(16,26),SP(16,26)  ADJU   2
0446         $,SD(16,26),ST(16,26),DU1,DU2,HS                            ADJU   3
0447          COMMON/ADJCOM/A(26,3), B(26), X(26), KOUNT                 ADJU   4
0448          DIMENSION PQ(26), QQ(26), UC(26), VC(26), WC(26), U(26), V(26),  ADJU   5
0449         $ W(26)                                                     ADJU   6
0450  C       ASSUMPTIONS@                                               ADJU   7
0451  C        HS IS THE SURFACE LEVEL                                   ADJU   8
0452  C        ALL DATA VALUES ABOVE SURFACE LEVEL ARE IN 1 KM INCREMENTS  ADJU   9
0453          E1=0.075                                                   ADJU  10
0454          E2=0.150                                                   ADJU  11
0455          MAXIT=3                                                    ADJU  12
0456          KSMAX=10                                                   ADJU  13
0457          HSJ = HS                                                   ADJU  14
0458          IF (HS.LT.0.) HSJ = 0.                                     ADJU  15
0459          JJ=INT(HSJ+2.)                                             ADJU  16
0460          STEST=0.05                                                 ADJU  17
0461          ISS=1                                                      ADJU  18
0462          CONST=28703./980.665                                       ADJU  19
0463          N=26                                                       ADJU  20
0464          ITER=0                                                     ADJU  21
0465          UC(1)=SQRT(ABS(SP(KOUNT,1)))                               ADJU  22
0466          VC(1)=SQRT(ABS(SD(KOUNT,1)))                               ADJU  23
0467          WC(1)=SQRT(ABS(ST(KOUNT,1)))                               ADJU  24
0468          DO 5 I=JJ,N                                                ADJU  25
0469          UC(I)=SQRT(ABS(SP(KOUNT,I)))                               ADJU  26
0470          VC(I)=SQRT(ABS(SD(KOUNT,I)))                               ADJU  27
0471        5 WC(I)=SQRT(ABS(ST(KOUNT,I)))                               ADJU  28
0472          NM=N-1                                                     ADJU  29
0473          NP=N+1                                                     ADJU  30
0474  C.....SETS UP QUADRATURE FACTORS                                   ADJU  31
0475          PQ(1)=500.*(FLOAT(INT(HSJ+1.))-HS)/(CONST*T(KOUNT,1))      ADJU  32
0476          QQ(1)=500.*(FLOAT(INT(HSJ+1.))-HS)/(CONST*T(KOUNT,JJ))     ADJU  33
0477          DO 15 I=JJ,NM                                              ADJU  34
0478          IP=I+1                                                     ADJU  35
0479          PQ(I)=500./(CONST*T(KOUNT,I))                              ADJU  36
0480       15 QQ(I)=500./(CONST*T(KOUNT,IP))                            ADJU  37
0481          GO TO 58                                                   ADJU  38
0482       12 NM=N-1                                                    ADJU  39
0483          NP=N+1                                                     ADJU  40
0484          DO 14 I=1,26                                               ADJU  41
0485          U(I)=UC(I)*UC(I)                                           ADJU  42
0486          V(I)=VC(I)*VC(I)                                           ADJU  43
0487          W(I)=WC(I)*WC(I)                                           ADJU  44
```

```
0488        14 CONTINUE                                                          ADJU  45
0489     C.....INITIALIZE A(I,J)                                                 ADJU  46
0490           DO 20 I=1,26                                                      ADJU  47
0491           DO 20 J=1,3                                                       ADJU  48
0492        20 A(I,J)=0.                                                         ADJU  49
0493     C.....SETS UP COEFFICIENTS                                              ADJU  50
0494           I2=0                                                             ADJU  51
0495           DO 35 I=1,NM                                                      ADJU  52
0496           IF(I.GT.1.AND.I.LT.JJ) GO TO 35                                   ADJU  53
0497           AW=1./SP(KOUNT,I)                                                 ADJU  54
0498           BW=1./SD(KOUNT,I)                                                 ADJU  55
0499           CW=1./ST(KOUNT,I)                                                 ADJU  56
0500           IM=I-1                                                           ADJU  57
0501           IF(I.EQ.JJ) IM=1                                                  ADJU  58
0502           IP=I+1                                                           ADJU  59
0503           IF (I.EQ.1) IP=JJ                                                 ADJU  60
0504           I2=I2+1                                                          ADJU  61
0505           AW1=1./SP(KOUNT,IP)                                               ADJU  62
0506           BW1=1./SD(KOUNT,IP)                                               ADJU  63
0507           CW1=1./ST(KOUNT,IP)                                               ADJU  64
0508           IF(I.EQ.1) GO TO 25                                               ADJU  65
0509           A(I2,1)=-(1.-QQ(IM))*(1.+PQ(I))/AW+(1./BW+1./CW)*PQ(I)*QQ(IM)     ADJU  66
0510        25 A(I2,2)=((1.-QQ(I))**2)/AW1+((1.+PQ(I))**2)/AW+(1./BW+1./CW)      ADJU  67
0511        $ *(PQ(I)**2)+(1./BW1+1./CW1)*QQ(I)**2                              ADJU  68
0512           IF(I.EQ.NM) GO TO 30                                              ADJU  69
0513           A(I2,3)=-(1.-QQ(I))*(1.+PQ(IP))/AW1+(1./BW1+1./CW1)*              ADJU  70
0514        $ PQ(IP)*QQ(IP)                                                     ADJU  71
0515        30 B(I2)=U(IP)-U(I)-(U(I)-V(I)+W(I))*PQ(I)-(U(IP)-V(IP)+W(IP))*QQ(I) ADJU  72
0516        35 CONTINUE                                                          ADJU  73
0517           CALL DIAGEQ(I2)                                                   ADJU  74
0518     C.....FINDS CORRECTIONS                                                 ADJU  75
0519           AW=1./SP(KOUNT,1)                                                 ADJU  76
0520           BW=1./SD(KOUNT,1)                                                 ADJU  77
0521           CW=1./ST(KOUNT,1)                                                 ADJU  78
0522           UC(1)=SQRT(ABS(U(1)+X(1)*(1.+PQ(1))/AW))                          ADJU  79
0523           VC(1)=SQRT(ABS(V(1)-X(1)*PQ(1)/BW))                               ADJU  80
0524           WC(1)=SQRT(ABS(W(1)+X(1)*PQ(1)/CW))                               ADJU  81
0525           AW=1./SP(KOUNT,N)                                                 ADJU  82
0526           BW=1./SD(KOUNT,N)                                                 ADJU  83
0527           CW=1./ST(KOUNT,N)                                                 ADJU  84
0528           UC(N)=SQRT(ABS(U(N)-X(I2)*(1.-QQ(NM))/AW))                        ADJU  85
0529           VC(N)=SQRT(ABS(V(N)-X(I2)*QQ(NM)/BW))                             ADJU  86
0530           WC(N)=SQRT(ABS(W(N)+X(I2)*QQ(NM)/CW))                             ADJU  87
0531           I2=1                                                             ADJU  88
0532           DO 40 I=JJ,NM                                                     ADJU  89
0533           I2=I2+1                                                          ADJU  90
0534           I2M=I2-1                                                         ADJU  91
0535           AW=1./SP(KOUNT,I)                                                 ADJU  92
0536           BW=1./SD(KOUNT,I)                                                 ADJU  93
0537           CW=1./ST(KOUNT,I)                                                 ADJU  94
0538           IM=I-1                                                           ADJU  95
0539           IF(I.EQ.JJ)IM=1                                                   ADJU  96
0540           UC(I)=ABS(U(I)      +(-X(I2M)*(1.-QQ(IM))+X(I2)*(1.+PQ(I)))/AW)   ADJU  97
0541           UC(I)=SQRT(UC(I))                                                 ADJU  98
0542           VC(I)=ABS(V(I)      -(X(I2M)*QQ(IM)+X(I2)*PQ(I))/BW)              ADJU  99
0543           VC(I)=SQRT(VC(I))                                                 ADJU 100
0544           WC(I)=ABS(W(I)      +(X(I2M)*QQ(IM)+X(I2)*PQ(I))/CW)              ADJU 101
0545        40 WC(I)=SQRT(WC(I))                                                 ADJU 102
0546     C.....GETS ADJUSTED VALUES                                             ADJU 103
0547     C.....    ADJUSTS TO TRIANGLE INEQUALITIES                             ADJU 104
0548        58 K=0                                                             ADJU 105
0549           DO 68 I=1,N                                                       ADJU 106
0550           IF(I.GT.1.AND.I.LT.JJ) GO TO 68                                   ADJU 107
0551           AU=UC(I)                                                         ADJU 108
```

```
0552            AV=VC(I)                                            ADJU 109
0553            AM=WC(I)                                            ADJU 110
0554            AMAX=AMAX1(AU,AV,AM)                                ADJU 111
0555            EE=E1*AMAX                                          ADJU 112
0556            EF=E2*AMAX                                          ADJU 113
0557            AW=SP(KOUNT,I)                                      ADJU 114
0558            BW=SD(KOUNT,I)                                      ADJU 115
0559            CW=ST(KOUNT,I)                                      ADJU 116
0560            COR=AU+AV-AM-EE                                     ADJU 117
0561            DIV=AW+BW+CW                                        ADJU 118
0562            IF(COR.GT.0.) GO TO 60                              ADJU 119
0563            COR=(AU+AV-AM-EF)/DIV                               ADJU 120
0564            AU=AU-COR*AW                                        ADJU 121
0565            AV=AV-COR*BW                                        ADJU 122
0566            AM=AM+COR*CW                                        ADJU 123
0567            GO TO 64                                            ADJU 124
0568         60 COR=AU-AV+AM-EE                                     ADJU 125
0569            IF(COR.GT.0.) GO TO 62                              ADJU 126
0570            COR=(AU-AV+AM-EF)/DIV                               ADJU 127
0571            AU=AU-COR*AW                                        ADJU 128
0572            AV=AV+COR*BW                                        ADJU 129
0573            AM=AM-COR*CW                                        ADJU 130
0574            GO TO 64                                            ADJU 131
0575         62 COR=-AU+AV+AM-EE                                    ADJU 132
0576            IF(COR.GT.0.) GO TO 66                              ADJU 133
0577            COR=(-AU+AV+AM-EF)/DIV                              ADJU 134
0578            AU=AU+COR*AW                                        ADJU 135
0579            AV=AV-COR*BW                                        ADJU 136
0580            AM=AM-COR*CW                                        ADJU 137
0581         64 K=K+1                                               ADJU 138
0582         66 UC(I)=AU                                            ADJU 139
0583            VC(I)=AV                                            ADJU 140
0584            WC(I)=AM                                            ADJU 141
0585         68 CONTINUE                                            ADJU 142
0586            KMAX=K                                              ADJU 143
0587        100 IF((ITER.EQ.0).OR.(KMAX.NE.0)) GO TO 110           ADJU 144
0588            GO TO 112                                           ADJU 145
0589        110 ITER=ITER+1                                        ADJU 146
0590            IF(ITER.LE.MAXIT) GO TO 12                         ADJU 147
0591        112 IF (ISS.NE.1) GO TO 999                            ADJU 148
0592        114 ITER=1                                             ADJU 149
0593            ISS=2                                               ADJU 150
0594            VTA=VC(1)                                           ADJU 151
0595            WTA=WC(1)                                           ADJU 152
0596            DO 120 I=JJ,NM                                      ADJU 153
0597            IM=I-1                                              ADJU 154
0598            IF(I.EQ.JJ)IM=1                                     ADJU 155
0599            VTB=VC(I)                                           ADJU 156
0600            WTB=WC(I)                                           ADJU 157
0601            VC(I)=(VC(I+1)+2.*VTB+VTA)*0.25                    ADJU 158
0602            WC(I)=(WC(I+1)+2.*WTB+WTA)*0.25                    ADJU 159
0603            VTA=VTB                                             ADJU 160
0604            WTA=WTB                                             ADJU 161
0605        120 CONTINUE                                            ADJU 162
0606            GO TO 12                                            ADJU 163
0607     C.....CALCULATE THE CORRECTED VARIANCES                   ADJU 164
0608        999 DO 1010 I=1,N                                      ADJU 165
0609            IF(I.GT.1.AND.I.LT.JJ) GO TO 1010                 ADJU 166
0610            SP(KOUNT,I)=UC(I)**2                               ADJU 167
0611            SD(KOUNT,I)=VC(I)**2                               ADJU 168
0612            ST(KOUNT,I)=WC(I)**2                               ADJU 169
0613       1010 CONTINUE                                            ADJU 170
0614            RETURN                                              ADJU 171
0615            END                                                ADJU 172
```

91

```
0616          SUBROUTINE SELEC4                                       SELE    1
0617          INTEGER IOTEM2                                          SELE    2
0618          COMMON/C4/XL(16),YL(16),NP,DUMMY(2499)                  SELE    3
0619   C                                                              SELE    4
0620   C      S                                                       SELE    5
0621   C      SUBROUTINE TO SELECT POINTS FOR INTERPOLATION           SELE    6
0622   C                                                              SELE    7
0623          COMMON /IOTEMP/ IOTEM1,IOTEM2,DUMMY2(62)                SELE    8
0624          COMMON /POINT/ IPT(16,5),LL(16),DXY(16,2)               SELE    9
0625          COMMON /ORDER/ IPTN(16,5),IREAD(65,3)                   SELE   10
0626   C                                                              SELE   11
0627          DIMENSION              IC(4),IL(2),JL(2) LIML(51),LIMU(51)  SELE   12
0628   C                                                              SELE   13
0629          DATA  LIML/15,14,13,12,11,10,9,8,7,6,5,4,3,2,23*1,2,3,4,5,6,7,8,9,  SELE   14
0630         110,11,12,13,14,15/                                      SELE   15
0631          DATA  LIMU/33,34,35,36,37,38,39,40,41,42,43,44,45,46,23*47,46,45,  SELE   16
0632         144,43,42,41,40,39,38,37,36,35,34,33/                    SELE   17
0633          DATA PI/3.14159/                                        SELE   18
0634   C                                                              SELE   19
0635   C                                                              SELE   20
0636   C          INITIALIZE                                          SELE   21
0637   C                                                              SELE   22
0638          PI4=PI/4.                                               SELE   23
0639          DEGRAD=PI/180.                                          SELE   24
0640          DO 1 I=1,16                                             SELE   25
0641          DO 1 J=1,5                                              SELE   26
0642        1 IPT(I,J)=0                                              SELE   27
0643   C                                                              SELE   28
0644   C          MAJOR LOOP FOR POINTS                               SELE   29
0645   C                                                              SELE   30
0646          DO 100 II=1,NP                                          SELE   31
0647   C                                                              SELE   32
0648          LA=ABS(XL(II))*10.+.5                                   SELE   33
0649          LO=YL(II)*10.+.5                                        SELE   34
0650          IF (LO.LT.0) LO = LO + 3600                             SELE  34B
0651          LL(II)=LA*10000+LO                                      SELE   35
0652          IF (XL(II).LT.0.) LL(II)=-LL(II)                        SELE   36
0653   C                                                              SELE   37
0654          IF (XL(II)-15.1) 15,30,30                               SELE   38
0655       15 IF (XL(II)) 50,40,40                                    SELE   39
0656   C                                                              SELE   40
0657   C          NMC GRID                                            SELE   41
0658   C                                                              SELE   42
0659       30 IPT(II,5)=2                                             SELE   43
0660          YEL = YL(II)                                            SELE  43B
0661          IF (YEL.LT.0.) YEL = YEL + 360.                         SELE  43C
0662          EL=(350.-YEL)*DEGRAD                                    SELE   44
0663          PHI=XL(II)*DEGRAD                                       SELE   45
0664          R=31.204359052*(SIN(PI4-PHI/2.)/COS(PI4-PHI/2.))        SELE   46
0665          XX=R*COS(EL)+24.                                        SELE   47
0666          YY=R*SIN(EL)+26.                                        SELE   48
0667          I=XX                                                    SELE   49
0668          J=YY                                                    SELE   50
0669          DX=XX-I                                                 SELE   51
0670          DY=YY-J                                                 SELE   52
0671          DXY(II,1)=DX                                            SELE   53
0672          DXY(II,2)=DY                                            SELE   54
0673          IF (XL(II).GT.17.18) GO TO 31                           SELE   55
0674          IF ((J.LT.1).OR.(J.GT.51)) GO TO 70                     SELE   56
0675          IF ((I.LT.LIML(J)).OR.(I.GT.LIMU(J))) GO TO 70          SELE   57
0676       31 IC(1)=I*1000+J                                          SELE   58
0677          IF ((ABS(DX).GT..1).OR.(ABS(DY).GT..1)) GO TO 32        SELE   59
0678          IP=1                                                    SELE   60
0679          GO TO 35                                                SELE   61
```

```
0680        32 CONTINUE                                                        SELE  62
0681           IF (XL(II).GT.17.18) GO TO 34                                   SELE  63
0682           IF (((I.GT.(LIMU(J)-1)).AND.((J.GE.15).AND.(J.LE.37)))          SELE  64
0683          1 .OR.(J.GT.50)) GO TO 70                                        SELE  65
0684           IF ((I+1.GT.LIMU(J+1)).OR.(I.LT.LIML(J+1))) GO TO 80            SELE  66
0685           IF ((I.EQ.LIMU(J)).OR.(I.EQ.LIML(J))) GO TO 80                  SELE  67
0686        34 IP=4                                                            SELE  68
0687           IC(2)=(I+1)*1000+J                                              SELE  69
0688           IC(3)=I*1000+J+1                                                SELE  70
0689           IC(4)=(I+1)*1000+J+1                                            SELE  71
0690        35 CONTINUE                                                        SELE  72
0691           REWIND (IOTEM2)                                                 SELE  73
0692           DO 38 IPG=1,1977                                               SELE  74
0693           READ(IOTEM2) IJ                                                 SELE  75
0694           DO 38 K=1,IP                                                    SELE  76
0695        38 IF(IC(K).EQ.IJ) IPT(II,K)=IPG                                   SELE  77
0696           GO TO 100                                                       SELE  78
0697   C                                                                       SELE  79
0698   C          EQUATORIAL GRID                                              SELE  80
0699   C                                                                       SELE  81
0700        40 IPT(II,5)=1                                                     SELE  82
0701           L1=XL(II)                                                       SELE  83
0702           L2=YL(II)                                                       SELE  84
0703           IF (L2.LT.0) L2 = L2 + 360                                      SELE 84B
0704           IL(1)=L1/5                                                      SELE  85
0705           IL(2)=IL(1)+1                                                   SELE  86
0706           JL(1)=(L2/5)+1                                                  SELE  87
0707           JL(2)=JL(1)-1                                                   SELE  88
0708           DO 45 K1=1,2                                                    SELE  89
0709           DO 45 K2=1,2                                                    SELE  90
0710           IF ((APS(XL(II)-IL(K1)*5).GT.0.1).OR.(ABS(YL(II)-JL(K2)*5).GT.0.1)SELE  91
0711          1 ) GO TO 45                                                     SELE  92
0712           IF (JL(K2).EQ.72) JL(K2)=0                                      SELE  93
0713           IPT(II,1)=JL(K2)*4+IL(K1)+1                                     SELE  94
0714           GO TO 100                                                       SELE  95
0715        45 CONTINUE                                                        SELE  96
0716           IF (JL(1).EQ.72) JL(1)=0                                        SELE  97
0717           IPT(II,1)=JL(1)*4+IL(1)+1                                       SELE  98
0718           IPT(II,2)=JL(2)*4+IL(1)+1                                       SELE  99
0719           IPT(II,3)=JL(1)*4+IL(2)+1                                       SELE 100
0720           IPT(II,4)=JL(2)*4+IL(2)+1                                       SELE 101
0721           GO TO 100                                                       SELE 102
0722   C                                                                       SELE 103
0723   C          SOUTHERN HEMISPHERE                                          SELE 104
0724   C                                                                       SELE 105
0725        50 IPT(II,5)=3                                                     SELE 106
0726           L1=XL(II)                                                       SELE 107
0727           L2=YL(II)                                                       SELE 108
0728           IF (L2.LT.0) L2 = L2 + 360                                      SELE108B
0729           IF (ABS(XL(II)).LT.85.0) GO TO 51                              SELE 109
0730           IPT(II,1)=1                                                     SELE 110
0731           IF (ABS(XL(II)+90.).LT.0.11) GO TO 100                          SELE 111
0732        51 CONTINUE                                                        SELE 112
0733           IL(1)=(L1/5)-1                                                  SELE 113
0734           JL(1)=(L2/5)+1                                                  SELE 114
0735           IL(2)=IL(1)+1                                                   SELE 115
0736           JL(2)=JL(1)-1                                                   SELE 116
0737           DO 52 K1=1,2                                                    SELE 117
0738           DO 52 K2=1,2                                                    SELE 118
0739           IF ((ABS(XL(II)-IL(K1)*5).GT.0.1).OR.(ABS(YL(II)-JL(K2)*5).GT.0.1)SELE 119
0740          1 ) GO TO 52                                                     SELE 120
0741           IF (JL(K2).EQ.72) JL(K2)=0                                      SELE 121
0742           IPT(II,1)=JL(K2)*17-IL(K1)+1                                    SELE 122
0743           IF (IL(K1).NE.0) GO TO 100                                      SELE 123
```

```
0744            IPT(II,1)=JL(K2)*4+1                         SELE 124
0745            IPT(II,5)=1                                  SELE 125
0746            GO TO 100                                    SELE 126
0747         52 CONTINUE                                     SELE 127
0748            IF (JL(1).EQ.72) JL(1)=0                     SELE 128
0749            IF (IPT(II,1).EQ.1) GO TO 54                 SELE 129
0750            IPT(II,1)=JL(1)*17-IL(1)+1                   SELE 130
0751            IPT(II,2)=JL(2)*17-IL(1)+1                   SELE 131
0752            IF (IL(2)) 55,53,55                          SELE 132
0753         53 IPT(II,3)=JL(1)*4+1                          SELE 133
0754            IPT(II,4)=JL(2)*4+1                          SELE 134
0755            IPT(II,5)=1133                               SELE 135
0756            GO TO 100                                    SELE 136
0757         54 IPT(II,2)=JL(1)*17-IL(2)+1                   SELE 137
0758            IPT(II,3)=JL(2)*17-IL(2)+1                   SELE 138
0759            IPT(II,5)=333                                SELE 139
0760            GO TO 100                                    SELE 140
0761         55 CONTINUE                                     SELE 141
0762            IPT(II,3)=JL(1)*17-IL(2)+1                   SELE 142
0763            IPT(II,4)=JL(2)*17-IL(2)+1                   SELE 143
0764            GO TO 100                                    SELE 144
0765      C                                                  SELE 145
0766      C     BODERLINE POINTS                            SELE 146
0767      C                                                  SELE 147
0768         70 CONTINUE                                     SELE 148
0769      C     TWO NMC, TWO EQUATORIAL                      SELE 149
0770            IPT(II,5)=2211                               SELE 150
0771            L=YL(II)                                     SELE 151
0772            IPT(II,1)=((L/5)+2)*4                        SELE 152
0773            IPT(II,2)=IPT(II,1)-4                        SELE 153
0774            IF (L.GE.355) IPT(II,1)=4                    SELE 154
0775      C                                                  SELE 155
0776            IF (J.LT.1) J=1                              SELE 156
0777            IF (J.GT.51) J=51                            SELE 157
0778            IF (I.LT.LIML(J)) I=LIML(J)                  SELE 158
0779            IF (I.GT.LIMU(J)) I=LIMU(J)                  SELE 159
0780            IC(1)=I*1000+J                               SELE 160
0781            IF ((J.LT.15).OR.(J.GT.37)) GO TO 72         SELE 161
0782            IC(2)=I*1000+J+1                             SELE 162
0783            GO TO 76                                     SELE 163
0784         72 IF ((J.NE.1).AND.(J.NE.51)) GO TO 74         SELE 164
0785            IF (I.EQ.LIMU(J)) GO TO 73                   SELE 165
0786            IC(2)=(I+1)*1000+J                           SELE 166
0787            GO TO 76                                     SELE 167
0788         73 IC(2)=(I-1)*1000+J                           SELE 168
0789            GO TO 76                                     SELE 169
0790         74 IF (I.EQ.LIML(J)) GO TO 75                   SELE 170
0791            IC(2)=LIMU(J+1)*1000+J+1                     SELE 171
0792            GO TO 76                                     SELE 172
0793         75 IC(2)=LIML(J+1)*1000+J+1                     SELE 173
0794      C                                                  SELE 174
0795         76 REWIND (IOTEM2)                              SELE 175
0796            DO 77 IPG=1,1977                             SELE 176
0797            READ(IOTEM2) IJ                              SELE 177
0798            DO 77 K=1,2                                  SELE 178
0799         77 IF(IC(K).EQ.IJ) IPT(II,K+2)=IPG              SELE 179
0800            GO TO 100                                    SELE 180
0801      C                                                  SELE 181
0802         80 CONTINUE                                     SELE 182
0803      C     THREE NMC, ONE EQUATORIAL                    SELE 183
0804            IPT(II,5)=2212                               SELE 184
0805            IC(2) = 0                                    SELE 185
0806            L=YL(II)                                     SELE 186
0807            IPT(II,2)=((L/5)+1)*4                        SELE 187
```

```
0808          IF (L.GE.355) IPT(II,2)=4                              SELE 188
0809          IF (I.EQ.LIML(J)) GO TO 84                             SELE 189
0810          IF (J.GT.37) GO TO 82                                  SELE 190
0811          IC(1)=I*1000+J                                         SELE 191
0812          IC(3)=I*1000+J+1                                       SELE 192
0813          IC(4)=(I+1)*1000+J+1                                   SELE 193
0814          GO TO 88                                               SELE 194
0815       82 IC(1)=(I+1)*1000+J                                     SELE 195
0816          IC(3)=I*1000+J                                         SELE 196
0817          IC(4)=I*1000+J+1                                       SELE 197
0818          GO TO 88                                               SELE 198
0819       84 IF (J.GT.37) GO TO 86                                  SELE 199
0820          IC(1)=(I-1)*?00+J+1                                    SELE 200
0821          IC(3)=I*1000+J+1                                       SELE 201
0822          IC(4)=I*1000+J                                         SELE 202
0823          GO TO 88                                               SELE 203
0824       86 IC(1)=(I+1)*1000+J+1                                   SELE 204
0825          IC(3)=(I+1)*1000+J                                     SELE 205
0826          IC(4)=I*1000+J                                         SELE 206
0827    C                                                            SELE 207
0828       88 REWIND (IOTEM2)                                        SELE 208
0829          DO 89 IPG=1,1977                                       SELE 209
0830          READ(IOTEM2) IJ                                        SELE 210
0831          DO 89 K=1,4                                            SELE 211
0832          IF(IC(K).EQ.0) GO TO 89                                SELE 212
0833          IF(IC(K).EQ.IJ) IPT(II,K)=IPG                         SELE 213
0834       89 CONTINUE                                               SELE 214
0835    C                                                            SELE 215
0836      100 CONTINUE                                               SELE 216
0837          DO 150 I=1,16                                          SELE 217
0838          DO 150 J=1,5                                           SELE 218
0839      150 IPTN(I,J)=IPT(I,J)                                     SELE 219
0840          CALL SORT4(NP)                                         SELE 220
0841          RETURN                                                 SELE 221
0842          END                                                    SELE 222
0843          SUBROUTINE INTRP4 (LALON)                              INTR   1
0844    C                                                            INTR   2
0845    C         SUBROUTINE TO INTERPOLATE VALUES                   INTR   3
0846    C                                                            INTR   4
0847          DIMENSION XLL(4),YLL(4),XC(4),YC(4)                    INTR   5
0848    C                                                            INTR   6
0849          COMMON/INT/D(208,5),IG(5),DXY(2),DLA(4),DLO(4)         INTR   7
0850    C                                                            INTR   8
0851          DEGRAD=3.14159/180.                                    INTR   9
0852          LALO=IABS(LALON)                                       INTR  10
0853          L1=LALO/10000                                          INTR  11
0854          L2=LALO-L1*10000                                       INTR  12
0855          XL=L1/10.                                              INTR  13
0856          YL=L2/10.                                              INTR  14
0857          IF (IG(5)-2) 30,20,10                                  INTR  15
0858       10 IF (IG(5)-3) 30,30,50                                  INTR  16
0859    C                                                            INTR  17
0860    C         INTERPOLATE FROM NMC GRID                          INTR  18
0861    C                                                            INTR  19
0862       20 CONTINUE                                               INTR  20
0863          DO 25 L=1,26                                           INTR  21
0864          DO 22 J=1,4                                            INTR  22
0865       22 IF (D(L,J).LT.0.01) GO TO 25                           INTR  23
0866          DO 24 K=1,8                                            INTR  24
0867          I=(K-1)*26+L                                           INTR  25
0868          D(I,5)=(1.-DXY(2))*((1.-DXY(1))*D(I,1)+DXY(1)*D(I,2))  INTR  26
0869        1 +DXY(2)*(((1.-DXY(1))*D(I,3))+DXY(1)*D(I,4))           INTR  27
0870       24 CONTINUE                                               INTR  28
0871       25 CONTINUE                                               INTR  29
```

```
0872          RETURN                                                    INTR  30
0873    C                                                               INTR  31
0874    C     INTERPOLATE FROM EQUATION FOR SOUTHERN HEMISPHERE GRID    INTR  32
0875    C                                                               INTR  33
0876       30 CONTINUE                                                  INTR  34
0877          DO 32 J=1,2                                               INTR  35
0878          XLL(J)=DLA(J)                                             INTR  36
0879          YLL(J)=DLO(J)                                             INTR  37
0880          IF ((YL.GE.355.).AND.(YLL(J).LT.0.01)) YLL(J)=360.        INTR  38
0881       32 CONTINUE                                                  INTR  39
0882          X=(YLL(1)-YL)/5.                                          INTR  40
0883          Y=(XL-XLL(1))/5.                                          INTR  41
0884          IF (IG(5).EQ.3) Y=-Y                                      INTR  42
0885          DO 38 L=1,26                                              INTR  43
0886          DO 36 J=1,4                                               INTR  44
0887       36 IF (D(L,J).LT.0.01) GO TO 38                              INTR  45
0888          DO 37 K=1,8                                               INTR  46
0889          I=(K-1)*26+L                                              INTR  47
0890          D(I,5)=D(I,1)+X*(D(I,2)-D(I,1))+Y*(D(I,3)-D(I,1))+X*Y*    INTR  48
0891         1 (D(I,4)-D(I,3)-D(I,2)+D(I,1))                            INTR  49
0892       37 CONTINUE                                                  INTR  50
0893       38 CONTINUE                                                  INTR  51
0894          RETURN                                                    INTR  52
0895    C                                                               INTR  53
0896    C         INTERPOLATE FROM ACROSS GRIDS                         INTR  54
0897    C                                                               INTR  55
0898       50 CONTINUE                                                  INTR  56
0899          IF (IG(5).NE.1133) GO TO 55                               INTR  57
0900          IG(5)=3                                                   INTR  58
0901          GO TO 30                                                  INTR  59
0902       55 CONTINUE                                                  INTR  60
0903          IF (IG(5).NE.333) GO TO 60                                INTR  61
0904          DLO(1)=(DLO(2)+DLO(3))/2.                                 INTR  62
0905          DO 52 I=1,208                                             INTR  63
0906       52 D(I,4)=D(I,3)                                             INTR  64
0907          DLA(4)=DLA(3)                                             INTR  65
0908          DLO(4)=DLO(3)                                             INTR  66
0909       60 CONTINUE                                                  INTR  67
0910          DO 62 I=1,4                                               INTR  68
0911          XLL(I)=DLA(I)                                             INTR  69
0912          YLL(I)=DLO(I)                                             INTR  70
0913          IF ((YL.GT.350.).AND.(YLL(I).LT.0.01)) YLL(I)=360.        INTR  71
0914       62 CONTINUE                                                  INTR  72
0915          ITH=0                                                     INTR  73
0916          X=YLL(1)-YL                                               INTR  74
0917          Y=YL-XLL(1)                                               INTR  75
0918       63 CONTINUE                                                  INTR  76
0919          DO 65 I=2,4                                               INTR  77
0920          XC(I)=YLL(1)-YLL(I)                                       INTR  78
0921       65 YC(I)=XLL(I)-XLL(1)                                       INTR  79
0922          TH2=3.14159/4                                             INTR  80
0923          TH3=3.14159/4                                             INTR  81
0924          IF (ABS(XC(2)).GT.0.01) TH2=ATAN(YC(2)/XC(2))             INTR  82
0925          IF (ABS(YC(3)).GT.0.01) TH3=ATAN(XC(3)/YC(3))             INTR  83
0926          IF (XC(2).LT.0.) TH2=3.14159+TH2                          INTR  84
0927          IF (XC(3).LT.0.) TH3=3.14159+TH3                          INTR  85
0928          DNN=COS(TH2+TH3)                                          INTR  86
0929          IF (ABS(DNN).GT.0.001) GO TO 66                           INTR  87
0930          ITH=ITH+1                                                 INTR  88
0931          IF (ITH.EQ.2) GO TO 66                                    INTR  89
0932          XLL(3)=XLL(4)                                             INTR  90
0933          YLL(3)=YLL(4)                                             INTR  91
0934          DO 61 I=1,208                                             INTR  92
0935       61 D(I,3)=D(I,4)                                             INTR  93
```

```
0936            GO TO 63                                    INTR  94
0937         66 CONTINUE                                    INTR  95
0938            ZA=SQRT(XC(2)**2+YC(2)**2)                  INTR  96
0939            IF (ITH.LT.2) GO TO 69                      INTR  97
0940            Z=SQRT(X**2+Y**2)                           INTR  98
0941            E=0.                                        INTR  99
0942            Z4=0.                                       INTR 100
0943            GO TO 71                                    INTR 101
0944         69 CONTINUE                                    INTR 102
0945            EB=SQRT(XC(3)**2+YC(3)**2)                  INTR 103
0946            Z4=(XC(4)*COS(TH3)-YC(4)*SIN(TH3))/DNN      INTR 104
0947            E4=(YC(4)*COS(TH2) XC(4)*SIN(TH2))/DNN      INTR 105
0948            Z=(X*COS(TH3)-Y*SIN(TH3))/DNN               INTR 106
0949            E=(Y*COS(TH2)-X*SIN(TH2))/DNN               INTR 107
0950            B=0.                                        INTR 108
0951            C=0.                                        INTR 109
0952            DD=0.                                       INTR 110
0953      C                                                 INTR 111
0954         71 CONTINUE                                    INTR 112
0955            DO 70 L=1,26                                INTR 113
0956            DO 68 J=1,4                                 INTR 114
0957         68 IF (D(L,J).LT.0.01) GO TO 70                INTR 115
0958            DO 67 K=1,8                                 INTR 116
0959            I=(K-1)*26+L                                INTR 117
0960            A=D(I,1)                                    INTR 118
0961            IF (ZA.GT.0.01) B=(D(I,2)-D(I,1))/ZA        INTR 119
0962            IF (EB.GT.0.01) C=(D(I,3)-D(I,1))/EB        INTR 120
0963            IF ((ABS(Z4).GT.0.01).AND.(ABS(E4).GT.0.01))INTR 121
0964          1 DD=(D(I,4)-A-B*Z4-C*E4)/(Z4*E4)            INTR 122
0965            D(I,5)=A+B*Z+C*E+DD*Z*E                     INTR 123
0966         67 CONTINUE                                    INTR 124
0967         70 CONTINUE                                    INTR 125
0968            RETURN                                      INTR 126
0969            END                                         INTR 127
0970            SUBROUTINE DIAGEQ(N)                        DIAG   1
0971      C     A(I,J)=DIAG. TERMS, I=ROW NO., J=DIAG. NO.  DIAG   2
0972      C     B(I)=RIGHT SIDE TERMS                       DIAG   3
0973      C     N=NO. OF ROWS                               DIAG   4
0974      C     K=NO. OF BORDER DIAGONALS, M=K+1=INDES OF PRIN. DIAG  DIAG   5
0975      C           2KH=TOTAL NO. OF DIAGS.               DIAG   6
0976      C     X(I)=SOLUTION                               DIAG   7
0977            COMMON/ADJCOM/A(26,3), B(26), X(26), KOUNT  DIAG   8
0978            K = 1                                       DIAG   9
0979            M=K+1                                       DIAG  10
0980            DO 30 L=1,N                                 DIAG  11
0981            ALM=A(L,M)                                  DIAG  12
0982            A(L,M)=1.                                   DIAG  13
0983            IF(L.EQ.N) GO TO 15                         DIAG  14
0984            I2=MINO(K,N-L)                              DIAG  15
0985            DO 10 I=1,I2                                DIAG  16
0986            MPI=M+I                                     DIAG  17
0987         10 A(L,MPI)=A(L,MPI)/ALM                       DIAG  18
0988         15 B(L)=B(L)/ALM                               DIAG  19
0989            IF(L.EQ.N) GO TO 30                         DIAG  20
0990            DO 25 I=1,I2                                DIAG  21
0991            LPI=L+I                                     DIAG  22
0992            FACT=A(LPI,M-I)                             DIAG  23
0993            DO 20 J=1,I2                                DIAG  24
0994            MJI=M+J-I                                   DIAG  25
0995         20 A(LPI,MJI)=A(LPI,MJI)-A(L,M+J)*FACT         DIAG  26
0996         25 B(LPI)=B(LPI)-B(L)*FACT                     DIAG  27
0997         30 CONTINUE                                    DIAG  28
0998            X(N)=B(N)                                   DIAG  29
0999            NM1=N-1                                     DIAG  30
```

97

```
1000          DO 50 L=1,NM1                               DIAG  31
1001          NML=N-L                                     DIAG  32
1002          SUM=0.                                      DIAG  33
1003          I2=MINO(K,L)                                DIAG  34
1004          DO 40 I=1,I2                                DIAG  35
1005       40 SUM=SUM+A(NML,M+I)*X(NML+I)                 DIAG  36
1006       50 X(NML)=B(NML)-SUM                           DIAG  37
1007          RETURN                                      DIAG  38
1008          END                                         DIAG  39
1009          SUBROUTINE SORT4(NP)                        SORT   1
1010    C                                                 SORT   2
1011    C          SORTS POINTS FOR SEQUENTIAL TAPE READING   SORT   3
1012    C                                                 SORT   4
1013    C       ASSIGNS POINT NUMBERS BY ORDER ON TAPE, NOT BY GRID   SORT   5
1014    C                                                 SORT   6
1015          COMMON /ORDER/ IPT (16,5),IREAD(65,3)       SORT   7
1016    C                                                 SORT   8
1017          DO 1 I=1,65                                 SORT   9
1018          DO 1 J=1,3                                  SORT  10
1019        1 IREAD(I,J)=0                                SORT  11
1020          DO 9 I=1,NP                                 SORT  12
1021          IF(IPT(I,5).LT.1) GO TO 10                  SORT  13
1022          IF(IPT(I,5).EQ.1) GO TO 9                   SORT  14
1023          IF(IPT(I,5).EQ.2) GO TO 2                   SORT  15
1024          IF(IPT(I,5).EQ.3) GO TO 4                   SORT  16
1025          IF(IPT(I,5).EQ.1133)GO TO 6                 SORT  17
1026          IF(IPT(I,5).EQ.2211) GO TO 7                SORT  18
1027          IF(IPT(I,5).EQ.2212)GO TO 8                 SORT  19
1028          IF (IPT(I,5).EQ.333) GO TO 4                SORT  20
1029          GO TO 10                                    SORT  21
1030        2 DO 3 J=1,4                                  SORT  22
1031          IF(IPT(I,J).LT.1) GO TO 3                   SORT  23
1032          IPT(I,J)=IPT(I,J)+288                       SORT  24
1033        3 CONTINUE                                    SORT  25
1034          GO TO 9                                     SORT  26
1035        4 DO 5 J=1,4                                  SORT  27
1036          IF(IPT(I,J).LT.1) GO TO 5                   SORT  28
1037          IPT(I,J)=IPT(I,J)+2265                      SORT  29
1038        5 CONTINUE                                    SORT  30
1039          GO TO 9                                     SORT  31
1040        6 IF(IPT(I,1).GT.0)IPT(I,1)=IPT(I,1)+2265     SORT  32
1041          IF(IPT(I,2).GT.0)IPT(I,2)=IPT(I,2)+2265     SORT  33
1042          GO TO 9                                     SORT  34
1043        7 IF(IPT(I,3).GT.0)IPT(I,3)=IPT(I,3)+288      SORT  35
1044          IF(IPT(I,4).GT.0)IPT(I,4)=IPT(I,4)+288      SORT  36
1045          GO TO 9                                     SORT  37
1046        8 IF(IPT(I,1).GT.0)IPT(I,1)=IPT(I,1)+288      SORT  38
1047          IF(IPT(I,3).GT.0)IPT(I,3)=IPT(I,3)+288      SORT  39
1048          IF(IPT(I,4).GT.0)IPT(I,4)=IPT(I,4)+288      SORT  40
1049        9 CONTINUE                                    SORT  41
1050    C                                                 SORT  42
1051    C          REORDERS POINT NUMBERS FOR READ        SORT  43
1052    C                                                 SORT  44
1053       10 IP=0                                        SORT  45
1054          DO 13 K=1,NP                                SORT  46
1055          DO 13 L=1,4                                 SORT  47
1056          MP=IPT(K,1)                                 SORT  48
1057          IF(MP.LT.1) GO TO 13                        SORT  49
1058       11 II=K                                        SORT  50
1059          JJ=L                                        SORT  51
1060          DO 12 I=1,NP                                SORT  52
1061          DO 12 J=1,4                                 SORT  53
1062          IF (IPT(I,J).LT.1) GO TO 12                 SORT  54
1063          IF(IPT(I,J).GT.3490) GO TO 12               SORT  55
```

98

```
1064          IF(IPT(I,J).GE.MP) GO TO 12                          SORT  56
1065          II=I                                                 SORT  57
1066          JJ=J                                                 SORT  58
1067          MP=IPT(I,J)                                          SORT  59
1068       12 CONTINUE                                             SORT  60
1069          IF(IPT(II,JJ).GT.3490) GO TO 14                      SORT  61
1070          IR=IR+1                                              SORT  62
1071          IREAD(IR,1)=II                                       SORT  63
1072          IREAD(IR,2)=JJ                                       SORT  64
1073          IREAD(IR,3)=IPT(II,JJ)                               SORT  65
1074          IPT(II,JJ)=IPT(II,JJ)+9000                           SORT  66
1075          MP=IPT(K,L)                                          SORT  67
1076          IF(MP.GT.3490) GO TO 13                              SORT  68
1077          GO TO 11                                             SORT  69
1078       13 CONTINUE                                             SORT  70
1079       14 RETURN                                               SORT  71
1080          END                                                  SORT  72
1081    !!T72-
```

```
0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c-------------------------------------------------------------------------------
0010          subroutine GetStringWidth ( ChrStr , lngth , nchar , iwidth , ixchar )
0011    c-------------------------------------------------------------------------------
0012    c     determine the width [iwidth] of the character string [ChrStr] and the
0013    c     distance each character is offset from the beginning [ixchar].  The
0014    c     character string is also left justified.
0015
0016          integer*2          ixchar(*)
0017          integer*2          lngth
0018          integer*2          nchar
0019          integer*2          iwidth
0020          character*(*)      ChrStr
0021          character*1        onechr
0022
0023    c.....determine the number of characters in the string and left justify it
0024
0025          nchar  = NumChr ( ChrStr , lngth )
0026
0027    c.....determine the width of the string and the individual position of each
0028    c     character
0029
0030          if ( nchar.gt.0 ) then
0031             iwidth = 0
0032             do i = 1 , nchar
0033                ixchar(i) = iwidth
0034                onechr    = chrstr(i:i)
0035                iwidth    = iwidth + CharWidth ( %val(onechr) )
0036             end do
0037          else if ( nchar.eq.0 ) then
0038             iwidth = 0
0039          end if
0040
0041          return
0042          end
```

99

```
0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Initialize segment
0010
0011    !!S Initialize
0012    c-----------------------------------------------------------------------
0013          subroutine initialize
0014    c-----------------------------------------------------------------------
0015
0016    !!SETC USINGINCLUDES = FALSE
0017          implicit none
0018
0019    c.....common block definition files
0020
0021          include 'AppleMenu.inc'
0022          include 'FileMenu.inc'
0023          include 'EditMenu.inc'
0024          include 'MapMenu.inc'
0025          include 'MBar.inc'
0026          include 'Globals.inc'
0027
0028          integer*2 kSysEnvironsVersion /1/
0029          integer*2 os_err
0030
0031    c.....declare TrapAvailable routine as a logical function (I wrote it)
0032
0033          logical TrapAvailable
0034
0035    c.....set up a 1-byte integer for a 2-item enumerated type;
0036    c.....a value of "1" means the second of the two enumerations
0037
0038          integer*1 ToolTrap /1/
0039
0040    c-----------------------------------------------------------------------
0041
0042          gInBackground = .false.
0043
0044    c.....call LSC FORTRAN routine to initialize their data structures
0045
0046          call InitFORTRAN
0047
0048          os_err = SysEnvirons ( %ref(kSysEnvironsVersion), %val(gMac) )
0049          if( gMac.machineType < 0 ) call AlertUser   ! RLH must provide this subroutine
0050
0051    c.....see if WaitNextEvent is available
0052
0053          gHasWaitNextEvent = TrapAvailable ( TWaitNextEvent, ToolTrap )
0054
0055    c.....set up a handle value in the menuHandle record
0056
0057          MenuBar.menuH = GetNewMBar ( %val(MenuBarID) )
0058          call SetMenuBar   ( %val(MenuBar.menuH) )
0059          call DisposHandle ( %val(MenuBar.menuH) )
0060
0061    c.....set up the menu handles for the menus
0062
0063          AppleMenuHndl.menuH = GetMHandle ( %val(AppleMenuID) )
0064          FileMenuHndl.menuH  = GetMHandle ( %val(FileMenuID) )
```

```
0065            EditMenuHndl.menuH  = GetMHandle ( %val(EditMenuID) )
0066            MapMenuHndl.menuH   = GetMHandle ( %val(MapMenuID) )
0067
0068    c.....add desk accessories to the Apple Menu
0069
0070            call AddResMenu ( %val( AppleMenuHndl ), %val( 'DRVR' ) )
0071
0072    c.....draw the menu bar
0073
0074            call DrawMenuBar
0075
0076            return
0077            end



0001    !!s LngDat
0002    c--------------------------------------------------------------------------
Segment LngDat
0003            block data LngDat
0004    c--------------------------------------------------------------------------
0005    c       array of longitude values
0006
0007            include 'LngCom.inc'
0008
0009            data (Longitude(i),i=    1,    80)/
0010    .-128.138,-128.654,-128.769,-129.544,-130.079,-130.481,-130.130,-129.754,
0011    .-130.129,-130.006,-141.010,-139.591,-138.504,-137.337,-135.975,-134.596,
0012    .-134.116,-132.957,-132.889,-132.890,-131.579,-130.247,-130.839,-132.135,
0013    .-132.882,-131.805,-130.525,-129.357,-128.856,-128.856,-127.615,-126.750,
0014    .-125.934,-125.206,-124.425,-123.125,-122.745,-122.186,-122.578,-122.752,
0015    .-122.854,-123.628,-123.533,-123.534,-123.847,-124.271,-124.618,-124.860,
0016    .-125.573,-125.515,-126.161,-126.177,-126.178,-126.972,-127.691,-127.495,
0017    .-126.688,-127.497,-127.389,-126.984,-127.611,-128.185,-128.139,-130.005,
0018    .-130.055,-130.327,-130.872,-131.023,-131.916,-131.976,-132.410,-133.169,
0019    .-133.453,-133.760,-133.769,-133.769,-134.687,-135.131,-135.364,-135.274/
0020            data (Longitude(i),i=   81,   160)/
0021    .-135.049,-135.889,-136.489,-136.081,-137.018,-137.769,-138.427,-139.235,
0022    .-139.852,-139.852,-139.055,-140.043,-141.043,-141.943,-142.946,-143.933,
0023    .-144.756,-145.703,-146.633,-147.432,-147.726,-147.726,-148.301,-148.125,
0024    .-148.626,-149.609,-150.297,-151.232,-151.830,-151.467,-151.400,-150.578,
0025    .-149.550,-149.467,-150.439,-151.355,-151.743,-151.743,-152.315,-152.628,
0026    .-153.311,-154.122,-153.475,-153.917,-154.326,-155.163,-155.734,-155.737,
0027    .-156.553,-157.044,-157.864,-158.559,-159.005,-159.684,-160.565,-161.410,
0028    .-162.187,-163.065,-163.329,-163.329,-162.545,-162.069,-161.261,-160.444,
0029    .-160.334,-159.623,-158.869,-158.325,-157.459,-157.623,-157.495,-156.981,
0030    .-156.779,-156.779,-157.462,-158.434,-159.397,-160.300,-161.231,-162.148/
0031            data (Longitude(i),i=  161,   240)/
0032    .-161.633,-161.786,-162.208,-162.225,-162.483,-163.378,-164.273,-164.647,
0033    .-164.648,-164.940,-163.909,-164.879,-165.061,-166.114,-165.731,-165.289,
0034    .-164.845,-164.122,-163.112,-162.801,-162.803,-161.810,-160.839,-160.945,
0035    .-160.787,-161.972,-162.768,-163.788,-164.944,-166.109,-166.705,-167.424,
0036    .-167.846,-167.846,-166.785,-165.582,-164.761,-163.887,-162.678,-161.538,
0037    .-162.189,-161.054,-161.802,-163.039,-163.883,-164.690,-164.764,-164.756,
0038    .-165.867,-166.235,-165.072,-163.767,-163.067,-162.515,-161.291,-160.172,
0039    .-159.770,-159.770,-159.934,-158.446,-157.184,-155.644,-154.586,-153.069,
0040    .-152.216,-152.216,-151.316,-149.838,-148.441,-147.024,-145.578,-144.098,
0041    .-142.628,-141.494,-141.010, -67.240, -66.401, -65.335, -64.039, -64.687/
0042            data (Longitude(i),i=  241,   320)/
0043    . -63.403, -63.822, -63.816, -62.579, -61.419, -61.887, -63.062, -62.595,
0044    . -63.500, -63.499, -63.302, -63.511, -64.404, -65.307, -64.798, -65.480,
0045    . -65.480, -66.731, -67.936, -67.282, -68.447, -68.070, -67.128, -66.736,
0046    . -66.698, -66.698, -65.715, -65.070, -64.507, -65.079, -65.269, -65.269,
0047    . -66.379, -67.242, -67.858, -68.992, -68.993, -68.995, -68.359, -67.511,
0048    . -66.733, -66.149, -67.116, -68.123, -69.121, -69.795, -70.873, -70.829,
```

101

```
0049        . -70.832, -71.775, -72.195, -73.081, -73.609, -74.622, -74.648, -74.649,
0050        . -74.673, -75.747, -76.648, -77.785, -78.066, -77.462, -76.264, -75.602,
0051        . -75.090, -74.716, -74.711, -73.501, -74.079, -73.872, -72.898, -71.671,
0052        . -71.184, -69.961, -69.227, -69.227, -70.422, -71.664, -72.915, -72.275/
0053        data (Longitude(i),i=  321,   400)/
0054        . -72.621, -72.941, -78.501, -77.770, -77.158, -76.658, -76.523, -76.549,
0055        . -76.760, -77.145, -77.863, -78.076, -78.074, -78.541, -77.884, -77.541,
0056        . -77.514, -77.641, -77.885, -77.885, -77.743, -78.033, -78.124, -77.162,
0057        . -76.143, -75.093, -74.023, -73.017, -72.795, -72.798, -71.787, -71.557,
0058        . -70.580, -69.542, -69.654, -69.622, -70.612, -71.032, -71.031, -70.050,
0059        . -69.700, -69.483, -68.538, -68.225, -68.794, -69.369, -69.369, -68.536,
0060        . -68.178, -67.678, -67.035, -66.460, -66.091, -66.023, -65.567, -65.491,
0061        . -65.488, -64.897, -64.156, -63.871, -63.125, -62.989, -62.990, -63.393,
0062        . -62.574, -61.991, -61.894, -61.346, -61.346, -62.138, -61.601, -61.145,
0063        . -60.393, -60.531, -59.777, -59.775, -59.827, -59.260, -58.756, -57.896/
0064        data (Longitude(i),i=  401,   480)/
0065        . -58.250, -59.056, -59.769, -60.598, -60.151, -59.458, -59.024, -59.024,
0066        . -58.372, -57.551, -57.127, -56.307, -55.746, -56.004, -55.793, -56.142,
0067        . -56.764,-122.747,-121.322,-120.264,-118.907,-117.742,-116.365,-114.978,
0068        .-114.076,-115.143,-115.114,-115.114,-113.816,-112.497,-111.222,-109.890,
0069        .-108.927,-107.943,-107.938,-107.200,-107.735,-107.891,-106.757,-106.756,
0070        .-105.722,-107.051,-108.290,-108.156,-106.855,-105.487,-105.040,-103.857,
0071        .-102.656,-101.346,-100.588,-100.588, -99.277, -97.957, -98.424, -97.283,
0072        . -96.747, -96.436, -95.490, -95.803, -95.327, -95.558, -95.443, -88.365,
0073        . -87.846, -86.635, -86.317, -85.843, -85.198, -85.432, -85.511, -85.584,
0074        . -85.586, -84.126, -82.727, -82.068, -81.310, -82.254, -82.125, -81.349/
0075        data (Longitude(i),i=  481,   560)/
0076        . -81.495, -82.154, -82.170, -83.201, -83.886, -85.155, -84.177, -85.140,
0077        . -86.269, -85.980, -86.710, -87.682, -88.704, -88.412, -88.412, -89.516,
0078        . -90.747, -89.653, -88.833, -87.638, -87.299, -87.866, -88.649, -89.786,
0079        . -90.124, -90.124, -90.156, -91.283, -92.389, -93.413, -92.424, -91.424,
0080        . -90.677, -91.411, -92.356, -92.479, -68.183, -68.827, -69.286, -69.977,
0081        . -70.711, -69.996, -69.981, -70.505, -70.979, -71.299, -71.299, -70.650,
0082        . -70.129, -69.675, -69.193, -68.604, -67.955, -67.709, -84.100, -83.224,
0083        . -82.378, -82.318, -82.253, -82.205, -82.296, -81.986, -81.989, -81.542,
0084        . -80.973, -80.530, -80.887, -80.141, -79.543, -79.333, -79.332, -79.740,
0085        . -79.411, -78.944, -78.961, -78.570, -78.725, -78.936, -78.919, -78.920/
0086        data (Longitude(i),i=  561,   640)/
0087        . -79.062, -79.346, -79.761, -78.957, -78.500, -56.764, -57.552, -58.306,
0088        . -58.997, -59.463, -60.005, -60.783, -61.563, -61.590, -61.589, -62.359,
0089        . -63.148, -63.935, -64.719, -65.501, -66.276, -66.959, -67.229, -67.924,
0090        . -68.183, -67.709, -67.004, -66.299, -65.537, -64.781, -64.212, -64.539,
0091        . -65.126, -65.866, -66.596, -66.848, -66.849, -66.099, -65.476, -64.739,
0092        . -64.979, -64.875, -64.579, -63.895, -63.772, -63.772, -63.304, -62.591,
0093        . -61.931, -61.364, -61.650, -61.882, -61.884, -62.542, -63.200, -63.859,
0094        . -64.436, -64.884, -65.408, -65.757, -65.756, -66.161, -66.009, -65.454,
0095        . -64.934, -64.221, -63.567, -64.273, -64.536, -64.574, -92.470, -93.408,
0096        . -93.950, -94.166, -94.556, -94.764, -94.788, -94.803, -94.291, -93.357/
0097        data (Longitude(i),i=  641,   720)/
0098        . -93.166, -93.158, -92.955, -92.696, -92.426, -91.530, -90.609, -89.813,
0099        . -88.927, -88.247, -87.664, -86.865, -86.047, -85.338, -84.492, -84.098,
0100        . -64.574, -65.112, -65.725, -66.402, -67.112, -67.221, -67.877, -68.053,
0101        . -68.053, -68.754, -69.191, -69.807, -70.362, -70.730, -70.881, -70.623,
0102        . -70.064, -69.936, -69.941, -70.505, -71.164, -71.803, -72.467, -73.117,
0103        . -73.719, -74.208, -74.258, -74.248, -74.019, -74.299, -74.669, -75.299,
0104        . -75.413, -75.420, -75.191, -75.041, -75.041, -75.371, -75.602, -75.950,
0105        . -75.876, -75.820, -76.303, -76.214, -75.841, -76.481, -76.507, -76.453,
0106        . -77.076, -77.251, -77.246, -76.653, -76.310, -76.849, -76.427, -76.287,
0107        . -76.896, -76.509, -75.914, -75.851, -75.851, -75.674, -75.882, -76.195
0108        data (Longitude(i),i=  721,   800)/
0109        . -76.138, -76.139, -76.635, -76.759, -77.247, -77.437, -77.434, -77.732,
0110        . -78.062, -78.663, -79.093, -79.340, -79.342, -79.757, -80.261, -80.856,
0111        . -80.897, -81.154, -81.490, -81.489, -81.493, -81.392, -81.279, -81.082,
0112        . -80.836, -80.549, -80.524, -80.448, -80.447, -80.621, -80.487, -80.291,
```

```
0113        .  -80.072,  -80.060,  -80.120,  -80.343,  -80.398,  -80.399,  -80.955,  -81.229,
0114        .  -81.621,  -81.845,  -81.777,  -81.778,  -82.242,  -82.579,  -82.408,  -82.713,
0115        .  -82.679,  -83.045,  -83.153,  -83.154,  -83.494,  -83.923,  -84.483,  -85.014,
0116        .  -85.504,  -85.978,  -86.453,  -86.453,  -87.019,  -87.598,  -88.007,  -88.417,
0117        .  -88.997,  -89.528,  -90.070,  -90.304,  -90.304,  -89.756,  -89.512,  -89.149,
0118        .  -89.410,  -89.410,  -89.778,  -90.354,  -90.899,  -91.338,  -91.338,  -91.664/
0119        data (Longitude(i),i=  801,   880)/
0120        .  -92.205,  -92.778,  -93.333,  -93.885,  -93.886,  -94.414,  -94.964,  -95.054,
0121        .  -95.457,  -95.956,  -96.526,  -96.800,  -96.800,  -97.183,  -97.292,  -97.723,
0122        .  -97.539,  -97.373,  -97.166,  -97.165,  -97.137,-117.122,-117.281,-117.282,
0123        .-117.613,-118.083,-118.554,-119.154,-119.646,-120.253,-120.447,-120.447,
0124        .-120.654,-120.844,-121.294,-121.644,-121.941,-121.745,-121.745,-122.221,
0125        .-122.488,-122.349,-121.711,-122.331,-122.949,-123.035,-123.034,-123.424,
0126        .-123.707,-123.764,-124.073,-124.339,-124.096,-124.069,-124.243,-124.242,
0127        .-124.404,-124.501,-124.371,-124.125,-124.097,-124.042,-123.997,-123.995,
0128        .-123.945,-123.952,-123.307,-123.992,-123.752,-123.751,-124.159,-124.348,
0129        .-124.672,-124.712,-124.713,-124.025,-123.279,-122.612,-123.056,-122.565/
0130        data (Longitude(i),i=  881,   960)/
0131        .-122.528,-122.330,-122.186,-128.535,-128.881,-128.540,-129.071,
0132        .-128.881,-128.824,-128.824,-129.231,-129.231,-130.176,-129.592,-130.286,
0133        .-130.176,-130.391,-129.778,-130.498,-130.391,-130.272,-130.272,-130.336,
0134        .-130.336,-130.165,-130.165,-131.783,-131.828,-131.260,-131.943,-132.496,
0135        .-131.783,-131.663,-131.916,-132.166,-132.842,-133.115,-132.309,-132.600,
0136        .-132.044,-131.663,-139.313,-139.313,-135.936,-135.012,-134.671,-135.936,
0137        .-127.470,-128.242,-127.553,-127.907,-127.121,-126.603,-126.068,-125.621,
0138        .-124.869,-124.809,-124.808,-125.180,-124.532,-123.831,-123.583,-124.023,
0139        .-124.717,-125.127,-125.494,-126.256,-127.024,-127.477,-123.100,-123.100,
0140        .-122.896,-122.896,-123.577,-123.577,-123.675,-123.675,-124.609,-124.609/
0141        data (Longitude(i),i=  961,  1040)/
0142        .-124.983,-124.983,-125.245,-125.245,-125.273,-125.273,-126.252,-126.252,
0143        .-126.127,-126.127,-127.998,-127.998,-127.953,-127.953,-127.240,-127.699,
0144        .-127.271,-127.240,-127.962,-127.962,-128.148,-128.148,-128.373,-128.373,
0145        .-126.768,-126.768, -64.383, -64.383, -65.068, -65.068, -64.660, -64.660,
0146        .  -70.588,  -70.588,  -73.412,  -74.746,  -73.412,  -73.412,  -75.010,  -75.037,
0147        .  -75.879,  -77.147,  -77.204,  -76.469,  -75.130,  -75.010,  -77.625,  -77.625,
0148        .  -76.571,  -76.571,  -77.501,  -78.564,  -77.496,  -77.501,  -82.186,  -82.411,
0149        .  -83.360,  -83.559,  -82.541,  -82.186,  -79.541,  -79.454,  -80.027,  -79.541,
0150        .  -78.661,  -78.661,  -79.127,  -79.107,  -79.973,  -79.505,  -79.127,  -79.556,
0151        .  -79.556,  -78.227,  -78.227,  -74.481,  -74.481,  -67.949,  -67.949,  -64.640/
0152        data (Longitude(i),i= 1041,  1120)/
0153        .  -64.640,  -61.355,  -61.355,  -57.941,  -57.941,-119.734,-118.123,-116.623,
0154        .-115.491,-116.614,-117.938,-119.186,-120.169,-120.395,-121.793,-122.892,
0155        .-123.701,-124.877,-125.516,-125.245,-125.264,-124.479,-124.396,-124.245,
0156        .-123.695,-121.849,-120.231,-119.745,-114.599,-113.321,-111.676,-111.677,
0157        .-110.085,-110.681,-109.112,-108.571,-108.232,-107.809,-107.806,-107.660,
0158        .-107.961,-108.274,-106.582,-105.446,-105.140,-104.801,-104.305,-104.414,
0159        .-104.414,-103.523,-102.218,-100.989,-102.293,-103.136,-101.857,-102.734,
0160        .-104.123,-104.266,-104.265,-105.546,-106.571,-107.257,-108.551,-109.743,
0161        .-110.009,-110.013,-111.379,-112.714,-113.618,-114.329,-115.757,-116.897,
0162        .-117.433,-117.431,-116.085,-114.617,-113.136,-111.645,-112.908,-114.378/
0163        data (Longitude(i),i= 1121,  1200)/
0164        .-115.871,-117.3  ,  118.415,-117.046,-116.816,-116.816,-115.364,-116.949,
0165        .-118.412,-118.709,-118.224,-116.993,-115.516,-114.048,-114.605,  -83.015,
0166        .  -82.042,  -81.592,  -80.532,  -80.936,  -82.012,  -82.904,  -83.872,  -84.572,
0167        .  -85.609,  -85.710,  -85.711,  -86.769,  -86.384,  -86.380,  -86.193,  -86.080,
0168        .  -85.014,  -83.898,  -83.022,-111.529,-111.529,-110.728,-110.728,-110.817,
0169        .-110.817,-109.883,-109.883,-109.577,-109.577,-109.447,-109.447,-108.951,
0170        .-108.951,-107.983,-107.983,-107.883,-107.883,-107.582,-107.582,-104.682,
0171        .-104.682,-101.694,-101.694,-100.086,-100.086,-100.324,-100.324,-100.096,
0172        .-100.096,-101.520,-101.520,-101.053,-101.053,-100.461,-100.461,-104.490,
0173        .-104.975,-106.343,-105.284,-104.490,-107.645,-107.645,  -97.875,  -96.917
0174        data (Longitude(i),i= 1201,  1280)/
0175        .  -96.234,  -96.079,  -97.447,  -98.634,  -98.500,  -98.221,  -97.875,  -96.136,
0176        .  -96.136,  -95.489,  -95.489,  -86.435,  -86.435,  -83.577,  -83.577,  -82.936,
```

```
0177        . -82.936, -83.348, -83.348, -83.921, -83.921, -84.910, -84.910, -90.620,
0178        . -90.620, -90.720, -90.720, -55.617, -55.975, -56.100, -56.504, -56.840,
0179        . -56.238, -55.492, -55.491, -56.070, -55.348, -54.574, -53.817, -53.457,
0180        . -53.456, -53.978, -54.134, -53.438, -53.941, -53.830, -53.227, -52.917,
0181        . -52.920, -53.177, -52.808, -52.965, -53.637, -53.920, -54.250, -54.487,
0182        . -55.129, -55.386, -55.385, -55.269, -55.988, -56.721, -57.468, -58.209,
0183        . -58.429, -58.431, -59.170, -59.139, -58.604, -58.520, -57.859, -57.997,
0184        . -57.996, -57.713, -57.438, -56.978, -56.636, -55.937, -55.615, -81.088/
0185        data (Longitude(i),i= 1281, 1360)/
0186        . -80.671, -81.479, -81.088, -79.297, -79.297, -63.492, -62.766, -62.115,
0187        . -62.002, -63.511, -64.077, -63.492, -55.467, -55.467, -54.530, -54.530,
0188        . -54.081, -54.081, -53.564, -53.564, -54.125, -54.125, -56.289, -56.289,
0189        . -60.309, -60.509, -60.980, -60.322, -60.229, -60.939, -61.517, -61.181,
0190        . -60.834, -60.309, -61.408, -61.951, -61.408, -63.985, -63.865, -63.156,
0191        . -62.432, -62.510, -63.138, -63.833, -64.413, -63.985, -64.483, -64.483,
0192        . -60.095, -60.095,-133.102,-132.733,-133.197,-133.102,-133.586,-132.900,
0193        .-132.346,-131.969,-132.117,-132.557,-133.124,-133.245,-133.586,-133.297,
0194        .-133.297,-133.286,-133.286,-131.238,-131.238,-131.467,-131.467,-131.819,
0195        .-131.819,-131.232,-130.982,-131.663,-131.232,-132.390,-132.390,-132.336/
0196        data (Longitude(i),i= 1361, 1440)/
0197        .-132.336,-132.703,-132.703,-132.802,-132.802,-133.052,-133.126,-133.793,
0198        .-133.052,-134.087,-133.875,-134.406,-134.087,-135.105,-134.734,-134.622,
0199        .-135.319,-135.571,-135.105,-135.700,-135.700,-135.738,-134.912,-135.862,
0200        .-135.033,-135.668,-136.228,-135.857,-135.738,-136.454,-136.454,-134.872,
0201        .-134.132,-133.792,-134.232,-133.930,-134.376,-134.645,-134.754,-134.872,
0202        .-145.781,-145.781,-146.097,-146.097,-146.938,-147.477,-147.199,-146.938,
0203        .-152.342,-152.342,-152.091,-152.766,-152.140,-152.091,-153.261,-152.379,
0204        .-153.026,-153.608,-154.521,-154.653,-153.778,-153.915,-153.261,-152.895,
0205        .-152.895,-153.876,-153.876,-153.457,-153.457,-154.083,-154.083,-154.421,
0206        .-154.421,-155.557,-155.557,-159.872,-159.872,-159.516,-159.516,-160.696/
0207        data (Longitude(i),i= 1441, 1520)/
0208        .-160.696,-162.233,-162.233,-162.754,-162.754,-163.493,-164.085,-164.908,
0209        .-164.494,-163.644,-163.493,-160.710,-160.710, -74.250, -74.250, -75.321,
0210        . -75.321, -75.714, -75.482, -75.693, -75.714, -75.798, -75.798, -76.053,
0211        . -76.053, -76.225, -76.225, -76.703, -76.703, -77.359, -77.359, -77.949,
0212        . -77.949, -80.432, -80.244, -80.444, -80.432, -80.187, -80.187, -80.268,
0213        . -80.268, -80.699, -80.699, -80.789, -80.789, -80.940, -80.940, -81.052,
0214        . -81.052, -81.413, -81.413, -81.492, -81.492, -81.547, -81.547, -81.654,
0215        . -81.654, -81.757, -81.757, -82.184, -82.184, -84.709, -84.709, -86.559,
0216        . -87.135, -86.556, -86.559, -88.095, -88.095, -88.553, -88.553, -88.820,
0217        . -88.820, -89.236, -89.236, -91.724, -91.724, -94.813, -94.813, -96.244/
0218        data (Longitude(i),i= 1521, 1600)/
0219        . -96.244, -96.404, -96.404, -96.839, -96.839, -97.053, -97.325, -97.076,
0220        . -97.053, -97.380, -97.320, -97.188, -97.338, -97.392, -97.380, -82.115,
0221        . -82.115, -66.765, -66.765, -68.241, -68.241, -70.019, -70.019, -70.557,
0222        . -70.557, -71.912, -72.514, -73.159, -73.803, -73.319, -72.655, -71.998,
0223        . -71.912, -81.657, -82.305, -82.989, -82.307, -81.657, -83.337, -83.337,
0224        . -83.625, -83.625, -83.921, -83.921, -85.502, -85.502, -86.987, -87.288,
0225        . -87.073, -86.987, -87.974, -87.974, -88.471, -89.068, -88.497, -88.471,
0226        . -87.767, -87.767, -90.601, -90.601, -90.531, -90.531, -90.413, -90.413,
0227        .-118.530,-118.530,-118.517,-118.517,-119.493,-119.493,-119.868,-119.868,
0228        .-120.109,-120.109,-120.328,-120.328,-122.588,-122.588,-123.492,-122.755/
0229        data (Longitude(i),i= 1601, 1680)/
0230        .-122.105,-123.260,-124.506,-123.464,-122.251,-121.176,-119.959,-120.270,
0231        .-119.157,-118.079,-117.994,-118.512,-119.667,-119.936,-120.826,-120.357,
0232        .-121.554,-122.399,-123.492, -98.431, -98.553, -98.110, -97.326, -97.283,
0233        . -96.944, -96.959, -96.356, -96.446, -96.695, -96.692, -97.069, -97.232,
0234        . -97.454, -97.814, -98.159, -98.916, -99.152, -98.548, -98.452,-115.027,
0235        .-114.009,-113.445,-112.434,-111.735,-111.104,-110.075,-111.131,-110.126,
0236        .-109.094,-109.233,-109.233,-110.320,-111.333,-111.914,-112.804,-113.851,
0237        .-114.782,-115.747,-115.166,-115.061,-115.851,-116.811,-116.054,-115.022,
0238        . -86.229, -86.259, -86.534, -87.008, -87.620, -87.816, -87.841, -87.865,
0239        . -75.812, -76.311, -76.214, -76.754, -77.440, -78.115, -78.799, -79.058/
0240        data (Longitude(i),i= 1681, 1760)/
```

104

```
0241    . -79.058, -79.748, -79.338, -78.711, -78.027, -77.328, -76.713, -76.047,
0242    . -75.812, -81.655, -81.014, -80.343, -79.772, -79.088, -79.452, -79.731,
0243    . -82.423, -82.575, -82.760, -83.406, -83.408, -83.301, -83.427, -84.084,
0244    . -84.698, -84.990, -84.990, -85.313, -85.482, -86.180, -86.349, -86.456,
0245    . -86.357, -86.230, -87.866, -87.796, -87.696, -87.497, -87.847, -87.429,
0246    . -87.067, -86.355, -86.354, -85.674, -84.951, -84.229, -84.306, -85.036,
0247    . -85.713, -86.146, -86.145, -86.806, -87.485, -88.108, -88.727, -89.311,
0248    . -89.993, -90.412, -90.413, -90.930, -91.609, -91.353, -90.835, -90.153,
0249    . -89.579, -89.580, -89.205, -88.549, -87.874, -87.151, -86.389, -86.330,
0250    . -86.329, -86.064, -85.394, -84.935, -84.776, -84.592, -83.938, -83.598/
0251    data (Longitude(i),i= 1761, 1840)/
0252    . -83.599, -82.891, -82.168, -81.476, -80.754, -80.395, -80.005, -79.660,
0253    . -79.660, -80.292, -80.893, -81.316, -81.295, -81.606, -81.723, -81.885,
0254    . -82.415, -83.102, -82.440, -83.114, -83.175, -79.731, -80.324, -80.952,
0255    . -81.533, -82.184, -82.854, -83.432, -83.175, -83.102, -82.536, -81.986,
0256    . -81.655,-112.766,-112.458,-112.198,-112.751,-112.953,-112.766, -35.816,
0257    . -34.735, -33.933, -33.612, -32.970, -32.035, -32.427, -31.704, -30.362,
0258    . -29.110, -28.356, -28.359, -27.012, -25.816, -25.146, -23.988, -22.807,
0259    . -24.014, -25.394, -26.848, -28.214, -26.931, -28.335, -29.236, -29.234,
0260    . -28.008, -26.614, -25.411, -26.815, -28.395, -28.429, -27.209, -25.735,
0261    . -24.566, -24.100, -22.882, -21.606, -21.742, -21.742, -21.763, -22.606/
0262    data (Longitude(i),i= 1841, 1920)/
0263    . -23.866, -25.456, -24.662, -26.334, -25.086, -26.750, -25.404, -25.739,
0264    . -25.689, -25.691, -24.561, -22.786, -21.070, -21.154, -22.479, -20.694,
0265    . -20.719, -21.440, -22.472, -22.476, -20.745, -19.578, -21.406, -22.244,
0266    . -20.928, -18.735, -18.286, -20.539, -20.489, -21.729, -21.727, -21.639,
0267    . -21.135, -19.915, -19.791, -17.916, -20.462, -17.595, -19.532, -16.467,
0268    . -14.935, -14.941, -13.031, -14.471, -17.818, -20.926, -22.522, -23.956,
0269    . -23.141, -22.095, -25.526, -27.370, -30.751, -27.880, -24.176, -21.320,
0270    . -22.016, -22.018, -24.965, -29.136, -33.046, -29.158, -31.653, -36.134,
0271    . -38.143, -42.303, -43.999, -43.987, -41.496, -39.814, -42.204, -43.365,
0272    . -44.367, -47.045, -49.752, -49.918, -53.087, -54.004, -55.989, -55.983/
0273    data (Longitude(i),i= 1921, 2000)/
0274    . -59.225, -57.507, -60.092, -60.842, -63.364, -65.386, -67.272, -64.934,
0275    . -63.787, -63.778, -64.932, -65.710, -68.273, -70.230, -72.526, -72.342,
0276    . -70.087, -68.390, -66.661, -66.662, -68.309, -66.171, -68.447, -70.726,
0277    . -69.839, -69.082, -67.326, -65.319, -63.412, -61.498, -60.119, -60.116,
0278    . -58.424, -58.239, -56.944, -56.143, -55.773, -55.258, -55.096, -55.098,
0279    . -54.676, -55.635, -55.415, -55.520, -53.946, -53.485, -52.219, -51.346,
0280    . -51.346, -52.551, -51.071, -50.860, -52.321, -53.794, -53.163, -52.011,
0281    . -50.542, -50.837, -50.434, -51.044, -52.412, -52.500, -67.231, -68.314,
0282    . -67.846, -69.096, -67.725, -68.594, -70.018, -70.024, -70.023, -68.604,
0283    . -67.181, -67.686, -68.958, -70.384, -69.483, -68.284, -69.728, -71.078/
0284    data (Longitude(i),i= 2001, 2080)/
0285    . -70.519, -71.967, -72.623, -72.623, -71.984, -72.134, -73.180, -73.745,
0286    . -74.787, -74.549, -75.894, -75.799, -77.392, -77.805, -76.997, -76.997,
0287    . -78.533, -77.907, -78.985, -80.470, -80.755, -80.583, -81.209, -82.783,
0288    . -84.284, -83.659, -83.636, -83.634, -85.323, -84.007, -85.592, -85.515,
0289    . -84.370, -85.981, -84.959, -84.803, -86.292, -84.973, -85.889, -86.431,
0290    . -86.703, -86.405, -86.404, -85.664, -86.931, -88.394, -89.252, -89.746,
0291    . -89.682, -89.992, -88.894, -87.459, -87.375, -89.332, -88.336, -86.844,
0292    . -85.871, -84.394, -82.930, -81.470, -80.009, -78.800, -78.800, -79.287,
0293    . -77.843, -77.616, -76.188, -75.594, -76.537, -75.152, -74.221, -73.137,
0294    . -72.941, -88.357, -87.833, -88.279, -89.644, -89.920, -90.618, -90.437/
0295    data (Longitude(i),i= 2081, 2160)/
0296    . -91.865, -92.001, -91.514, -91.511, -92.189, -92.876, -93.399, -94.501,
0297    . -95.833, -96.377, -96.410, -96.409, -96.203, -96.532, -95.595, -94.369,
0298    . -94.387, -93.402, -94.277, -95.431, -99.996, -98.270, -97.167, -98.348,
0299    . -97.023, -96.615, -97.007, -98.181, -98.738, -98.739, -99.614,-100.514,
0300    .-101.889,-102.715,-101.081,-100.228,-101.581,-100.477, -99.998,-112.427,
0301    .-111.422,-109.588,-109.431,-109.707,-108.396,-107.003,-105.677,-106.792,
0302    .-107.772,-107.772,-109.494,-111.094,-112.794,-114.429,-112.911,-111.087,
0303    .-113.024,-114.906,-116.773,-115.737,-116.506,-115.475,-113.639,-112.433,
0304    . -92.005, -90.217, -91.071, -89.167, -87.568, -85.583, -84.144, -82.115,
```

```
0305        . -80.256, -79.955, -79.331, -79.334, -80.753, -82.615, -84.496, -86.374/
0306        data (Longitude(i),i= 2161, 2240)/
0307        . -88.264, -90.125, -90.814, -90.817, -92.384, -92.036, -92.734, -94.743,
0308        . -96.403, -94.729, -93.260, -92.013, -76.248, -77.227, -73.447, -70.207,
0309        . -66.391, -63.226, -61.219, -63.625, -66.686, -68.990, -70.024, -70.023,
0310        . -67.208, -67.584, -69.281, -71.446, -73.745, -74.952, -77.667, -77.926,
0311        . -77.923, -75.308, -77.987, -75.975, -75.234, -76.490, -77.890, -79.861,
0312        . -81.738, -81.788, -81.783, -79.544, -77.872, -78.925, -80.638, -82.032,
0313        . -84.181, -85.978, -86.629, -86.629, -88.616, -89.107, -87.129, -88.181,
0314        . -85.856, -83.794, -82.723, -84.580, -85.530, -84.804, -86.688, -87.001,
0315        . -84.683, -82.543, -82.245, -82.248, -84.065, -84.684, -86.442, -85.897,
0316        . -83.267, -81.768, -82.658, -80.301, -78.025, -78.033, -78.552, -80.577/
0317        data (Longitude(i),i= 2241, 2320)/
0318        . -83.697, -86.578, -85.075, -87.174, -85.907, -89.081, -88.406, -87.291,
0319        . -87.288, -90.501, -90.610, -87.063, -85.502, -82.674, -79.578, -81.855,
0320        . -81.285, -77.439, -76.261, -94.334, -91.414, -89.816, -87.963, -87.344,
0321        . -85.020, -86.741, -87.969, -90.308, -92.790, -93.902, -91.830, -90.362,
0322        . -90.371, -92.951, -95.676, -96.599, -94.209, -96.027, -94.943, -94.342,
0323        . -87.354, -88.874, -89.331, -25.684, -25.900, -27.378, -26.568, -25.684,
0324        . -23.112, -21.930, -23.588, -24.473, -23.112, -21.904, -23.613, -21.917,
0325        . -21.904, -24.418, -22.944, -24.640, -24.418, -20.493, -20.493, -17.969,
0326        . -17.969, -45.139, -47.163, -44.777, -45.139, -51.699, -51.699, -54.302,
0327        . -52.996, -51.913, -52.783, -54.201, -54.872, -54.509, -54.302, -51.002/
0328        data (Longitude(i),i= 2321, 2400)/
0329        . -51.002, -80.772, -79.038, -77.319, -76.353, -77.904, -79.526, -80.143,
0330        . -80.798, -80.772, -77.114, -77.114, -78.459, -78.840, -78.471, -78.459,
0331        . -78.142, -78.142, -80.479, -80.479, -95.088, -93.327, -91.571, -90.725,
0332        . -91.477, -92.374, -94.058, -93.862, -95.161, -95.138, -95.653, -95.684,
0333        . -95.326, -95.088, -97.654, -99.141, -97.654, -94.827, -93.493, -93.439,
0334        . -95.344, -96.091, -94.827,-104.155,-104.155,-101.989,-101.989,-102.652,
0335        .-102.652,-104.134,-104.134,-100.297,-100.297,-105.649,-104.463,-106.023,
0336        .-105.649,-117.623,-118.222,-117.662,-117.623,-118.651,-116.488,-116.257,
0337        .-116.794,-118.944,-119.491,-121.549,-121.422,-120.200,-118.894,-118.651,
0338        .-114.190,-114.190,-109.792,-110.163,-112.057,-113.262,-111.364,-109.792/
0339        data (Longitude(i),i= 2401, 2480)/
0340        .-114.284,-114.284,-110.394,-111.410,-110.431,-110.394,-101.672,-101.672,
0341        .-103.703,-102.620,-100.143, -99.501,-101.969,-104.445,-103.412,-105.543,
0342        .-103.703, -93.175, -95.204, -93.175, -97.951, -96.009, -94.948, -97.039,
0343        . -97.868, -98.341, -97.951, -90.844, -90.844, -89.983, -89.983, -78.893,
0344        . -78.893, -99.733, -98.792, -99.733, -75.224, -75.224,-100.384, -98.214,
0345        . -97.518, -97.412, -97.640, -99.517,-100.531,-102.499,-101.478, -99.530,
0346        .-100.691,-100.384, -20.231, -20.231, -18.923, -18.629, -19.056, -18.923,
0347        . -17.675, -17.675, -19.314, -19.314, -19.194, -19.194, -17.721, -17.721,
0348        . -19.086, -19.086, -72.236, -72.236, -71.239, -71.239, -53.438, -53.438,
0349        . -53.584, -53.584, -52.971, -52.971, -90.195, -90.195, -90.692, -90.692/
0350        data (Longitude(i),i= 2481, 2560)/
0351        . -91.476, -91.476, -96.772, -96.772, -21.192, -21.192, -18.323, -18.323,
0352        . -18.152, -18.152, -54.971, -54.971, -35.818, -36.870, -37.427, -38.146,
0353        . -39.025, -40.214, -40.217, -40.256, -40.566, -41.567, -40.622, -40.865,
0354        . -41.749, -41.747, -41.538, -42.590, -42.242, -42.407, -43.079, -43.078,
0355        . -42.694, -42.976, -43.990, -43.127, -44.096, -45.088, -44.486, -45.173,
0356        . -44.774, -44.633, -44.635, -45.547, -45.466, -46.503, -47.519, -47.770,
0357        . -47.769, -48.699, -48.935, -49.701, -50.072, -49.701, -49.701, -50.769,
0358        . -51.382, -50.984, -50.176, -49.595, -49.586, -50.129, -50.902, -50.997,
0359        . -51.737, -51.941, -52.069, -51.533, -50.551, -50.551, -51.771, -52.986,
0360        . -52.041, -53.210, -52.700, -51.765, -50.566, -51.792, -52.701, -53.053,
0361        data (Longitude(i),i= 2561, 2640)/
0362        . -52.647, -51.370, -50.350, -50.349, -51.579, -52.873, -52.078, -50.848,
0363        . -52.035, -53.293, -52.287, -50.955, -50.770, -51.945, -52.491, -37.024,
0364        . -37.024, -37.191, -37.191, -37.987, -37.987, -40.447, -40.447, -41.036,
0365        . -41.036, -40.819, -40.819, -41.461, -41.461, -41.941, -41.941, -42.404,
0366        . -42.404, -42.215, -42.215, -42.248, -42.248, -43.818, -43.818, -43.366,
0367        . -43.366, -43.507, -43.507, -44.116, -44.116, -44.005, -44.005, -44.265,
0368        . -44.265, -45.152, -45.152, -44.931, -44.931, -45.008, -45.008, -45.438,
```

106

```
0369        . -45.438, -45.357, -45.357, -46.277, -46.277, -46.538, -46.538, -47.005,
0370        . -47.005, -46.330, -46.330, -46.207, -46.207, -47.084, -47.084, -47.082,
0371        . -47.082, -47.270, -47.270, -47.758, -47.758, -53.188, -53.188, -97.137/
0372        data (Longitude(i),i= 2641, 2720)/
0373        . -97.269, -97.517, -97.644, -97.716, -97.747, -97.741, -97.846, -97.737,
0374        . -97.408, -97.753, -97.604, -97.358, -97.179, -97.179, -96.867, -96.531,
0375        . -96.325, -96.044, -95.662, -95.134, -94.709, -94.388, -94.388, -93.881,
0376        . -93.373, -92.846, -92.357, -91.859, -91.334, -91.135, -90.734, -90.729,
0377        . -90.729, -90.574, -90.481, -90.375, -89.987, -89.463, -88.928, -88.419,
0378        . -87.876, -87.350, -87.113, -87.112, -86.815, -86.982, -87.338, -87.440,
0379        . -87.639, -87.624, -87.773, -88.030, -88.283, -88.283, -88.136, -88.169,
0380        . -88.226, -88.348, -88.745, -88.911, -88.911, -88.394, -88.213, -88.213,
0381        . -87.730, -87.219, -86.697, -86.184, -85.671, -85.671, -85.149, -84.630,
0382        . -84.244, -84.244, -83.860, -83.389, -83.132, -83.132, -83.229, -83.378/
0383        data (Longitude(i),i= 2721, 2800)/
0384        . -83.518, -83.518, -83.533, -83.528, -83.678, -83.733, -83.733, -83.735,
0385        . -83.806, -83.644, -83.644, -83.450, -83.142, -82.799, -82.561, -77.889,
0386        . -78.211, -78.431, -77.951, -77.739, -77.739, -78.102, -78.572, -78.988,
0387        . -78.977, -78.977, -79.463, -79.531, -79.531, -79.603, -79.603, -79.919,
0388        . -80.389, -80.380, -80.381, -80.042, -80.405, -80.908, -81.035, -81.215,
0389        . -81.215, -81.705, -82.179, -82.432, -82.432, -82.854, -82.896, -82.561,
0390        . -82.229, -81.794, -81.567, -81.567, -81.061, -80.615, -80.132, -80.031,
0391        . -80.037, -79.828, -79.829, -79.357, -78.872, -78.371, -78.027, -78.027,
0392        . -77.679, -77.366, -82.897, -83.113, -83.619, -83.603, -83.603, -83.963,
0393        . -84.432, -84.721, -85.170, -85.285, -85.285, -84.923, -85.426, -85.818/
0394        data (Longitude(i),i= 2801, 2880)/
0395        . -85.673, -85.690, -85.690, -86.071, -86.465, -86.753, -87.165, -87.099,
0396        . -87.099, -87.465, -87.302, -87.302, -87.612, -87.817, -87.817, -88.269,
0397        . -88.778, -89.060, -89.060, -89.556, -90.024, -90.096, -90.096, -90.579,
0398        . -91.097, -91.586, -92.007, -92.247, -94.528, -95.031, -95.416, -95.912,
0399        . -96.418, -96.932, -97.434, -97.934, -98.423, -98.871, -99.372, -99.858,
0400        .-100.350,-100.650,-100.650,-101.118,-101.573,-101.950,-102.477,-102.983,
0401        .-103.483,-103.802,-104.258,-104.729,-105.109,-105.378,-105.378,-105.618,
0402        .-105.234,-105.223,-105.433,-105.655,-105.756,-106.116,-106.483,-106.814,
0403        .-107.180,-107.614,-108.015,-108.307,-108.776,-109.258,-109.319,-109.425,
0404        .-109.805,-109.807,-110.162,-110.581,-110.840,-111.326,-111.727,-112.081/
0405        data (Longitude(i),i= 2881, 2960)/
0406        .-112.402,-112.632,-112.832,-113.091,-113.186,-113.636,-113.636,-114.192,
0407        .-114.702,-114.873,-114.686,-114.644,-114.392,-113.954,-113.589,-113.183,
0408        .-112.872,-112.736,-112.336,-112.012,-111.955,-111.955,-111.799,-111.402,
0409        .-111.290,-110.981,-110.702,-110.695,-110.162,-109.814,-109.451,-109.661,
0410        .-110.154,-110.499,-110.917,-111.361,-111.813,-112.097,-112.063,-112.209,
0411        .-112.309,-112.309,-112.756,-113.210,-113.742,-114.198,-114.620,-115.049,
0412        .-114.485,-114.113,-114.154,-114.587,-114.955,-115.189,-115.187,-115.646,
0413        .-115.793,-116.036,-116.304,-116.647,-116.855,-117.122,-117.122, -94.529,
0414        . -94.032, -93.583, -93.180, -92.766, -92.786, -92.247, -92.611, -92.786,
0415        . -64.748, -64.748, -72.330, -72.330, -71.586, -71.586, -71.848, -71.848/
0416        data (Longitude(i),i= 2961, 3040)/
0417        . -72.005, -72.005, -73.145, -73.145, -73.065, -73.065, -73.706, -73.189,
0418        . -73.657, -73.706, -73.814, -73.814, -74.344, -74.344, -73.852, -74.122,
0419        . -73.852, -74.284, -74.284, -75.308, -75.138, -75.308, -74.834, -74.834,
0420        . -74.530, -74.530, -75.627, -75.524, -75.627, -75.651, -75.651, -76.020,
0421        . -76.020, -77.701, -77.547, -77.701, -78.205, -77.854, -77.985, -78.437,
0422        . -78.157, -78.205, -77.880, -77.880, -77.506, -77.506, -76.735, -76.285,
0423        . -76.178, -76.314, -76.735, -77.950, -77.403, -77.038, -77.193, -77.232,
0424        . -77.513, -77.950, -78.963, -78.399, -78.959, -78.963, -67.938, -67.938,
0425        . -67.259, -66.737, -66.208, -65.689, -66.008, -66.538, -67.066, -67.259,
0426        . -65.562, -65.562, -64.862, -64.862, -65.013, -65.013, -64.640, -64.640,
0427        data (Longitude(i),i= 3041, 3120)/
0428        . -64.407, -64.407, -64.732, -64.732, -63.168, -63.168, -63.135, -63.135,
0429        . -62.862, -62.869, -62.960, -62.960, -62.839, -62.839, -62.600, -62.600,
0430        . -61.868, -61.868, -61.826, -61.826, -62.202, -62.202, -61.522, -61.522,
0431        . -61.757, -61.757, -61.288, -61.288, -61.468, -61.468, -61.199, -60.826,
0432        . -61.228, -61.199, -71.989, -71.492, -70.959, -70.446, -69.928, -69.519,
```

```
0433      . -69.112, -68.615, -68.446, -68.973, -69.505, -70.033, -70.523, -71.039,
0434      . -71.265, -71.757, -72.247, -72.775, -73.298, -73.819, -74.332, -73.894,
0435      . -73.364, -72.840, -72.346, -72.729, -72.786, -73.293, -72.868, -72.358,
0436      . -71.989, -68.784, -68.784, -71.522, -71.522, -73.707, -73.707, -73.798,
0437      . -73.798, -73.295, -72.803, -73.299, -73.295, -72.948, -72.948, -83.926/
0438      data (Longitude(i),i= 3121, 3200)/
0439      . -83.926, -83.959, -83.959, -81.395, -81.395, -80.112, -80.112, -78.373,
0440      . -77.910, -77.382, -76.864, -76.370, -76.859, -77.366, -77.856, -78.341,
0441      . -78.373, -84.924, -84.397, -84.215, -83.756, -83.257, -82.714, -82.193,
0442      . -81.647, -81.123, -80.592, -80.099, -79.631, -79.192, -78.648, -78.161,
0443      . -77.747, -77.219, -76.834, -76.303, -75.779, -75.493, -74.956, -74.529,
0444      . -74.132, -74.132, -74.645, -75.153, -75.686, -76.215, -76.745, -77.281,
0445      . -77.112, -77.541, -78.076, -78.488, -78.633, -79.173, -79.692, -80.203,
0446      . -80.697, -81.186, -81.719, -81.874, -82.420, -82.946, -83.356, -83.898,
0447      . -84.328, -84.868, -84.925, -79.632, -79.632, -79.089, -79.089, -78.493,
0448      . -78.493, -78.311, -78.311, -78.015, -78.015, -77.947, -77.947, -78.461/
0449      data (Longitude(i),i= 3201, 3280)/
0450      . -78.461, -78.823, -78.823, -78.966, -78.966, -79.077, -79.077, -79.226,
0451      . -79.226, -79.384, -79.384, -81.562, -81.562, -83.075, -82.576, -83.080,
0452      . -83.075,-106.248,-106.248,-106.472,-106.472,-109.009,-109.009,-109.916,
0453      .-109.916,-110.394,-110.394,-110.701,-110.701,-111.142,-111.142,-112.438,
0454      .-112.438,-112.604,-112.604,-112.886,-112.886,-113.589,-113.193,-113.589,
0455      .-114.651,-114.651,-118.304,-118.304,-115.260,-115.260,-112.134,-112.167,
0456      .-112.140,-112.134,-112.011,-112.011,-110.816,-110.816,-110.998,-110.998,
0457      .-114.772,-114.772,-106.687,-106.687,-106.649,-106.649, -91.539, -91.539,
0458      . -86.745, -86.745, -87.255, -87.255, -87.328, -87.328,-109.218,-109.217,
0459      . -87.998, -87.997, -87.935, -87.935, -86.986, -86.987, -86.630, -86.628/
0460      data (Longitude(i),i= 3281, 3360)/
0461      . -85.952, -85.952, -87.647, -87.648, -87.588, -87.591, -82.800, -82.799,
0462      . -83.051, -83.050, -85.593, -85.589, -82.266, -82.263, -82.224, -82.222,
0463      . -82.194, -82.192, -79.105, -79.104, -79.091, -79.090, -78.872, -78.870,
0464      . -81.087, -81.084, -81.740, -81.738, -81.811, -81.810, -82.316, -82.316,
0465      . -82.274, -82.274, -82.329, -82.329, -97.437, -97.560, -97.679, -97.740,
0466      . -97.708, -97.657, -97.539, -97.437, -85.773, -85.333, -84.974, -85.338,
0467      . -85.773, -85.924, -85.771, -79.918, -79.912, -79.687, -79.562, -77.367,
0468      . -77.049, -76.720, -76.309, -76.259, -78.596, -78.561, -78.065, -77.669,
0469      . -77.408, -77.129, -77.429, -77.319, -77.372, -77.509, -77.396, -77.356,
0470      . -76.259, -75.953, -75.642, -75.546, -75.286, -74.913, -74.406, -74.046/
0471      data (Longitude(i),i= 3361, 3440)/
0472      . -74.029, -74.031, -73.528, -73.061, -72.619, -72.188, -71.827, -71.309,
0473      . -71.325, -77.356, -77.699, -77.890, -78.811, -79.174, -79.644, -80.105,
0474      . -80.040, -80.340, -80.489, -80.870, -80.813, -80.801, -80.800, -80.914,
0475      . -80.539, -80.055, -79.844, -80.103, -80.338, -80.337, -80.729, -81.054,
0476      . -81.319, -81.115, -80.959, -81.174, -80.840, -80.399, -79.975, -79.696,
0477      . -79.464, -79.158, -79.136, -79.136, -78.848, -78.666, -78.435, -78.248,
0478      . -78.044, -77.771, -77.664, -77.277, -77.134, -76.789, -76.520, -76.265,
0479      . -76.262, -76.271, -76.272, -76.142, -75.821, -75.419, -75.084, -74.621,
0480      . -74.160, -73.732, -73.257, -72.790, -72.306, -71.896, -71.434, -71.166,
0481      . -70.738, -70.404, -78.596, -79.018, -78.811, -70.404, -70.332, -70.267/
0482      data (Longitude(i),i= 3441, 3520)/
0483      . -70.147, -70.162, -70.161, -70.094, -70.146, -70.247, -70.298, -70.594,
0484      . -70.473, -70.493, -73.912, -73.867, -73.586, -73.019, -72.350, -72.815,
0485      . -72.478, -72.862, -72.908, -73.051, -73.282, -72.735, -72.589, -72.590,
0486      . -72.866, -73.336, -73.567, -73.635, -73.342, -73.342, -73.754, -74.412,
0487      . -74.926, -74.926, -75.536, -74.887, -74.152, -74.113, -74.686, -73.986,
0488      . -73.263, -74.008, -74.029, -74.055, -74.056, -74.401, -73.870, -73.884,
0489      . -74.654, -74.068, -73.803, -73.662, -73.465, -73.384, -73.387, -72.626,
0490      . -73.241, -72.509, -72.474, -73.292, -73.645, -73.215, -72.724, -71.942,
0491      . -71.475, -71.475, -72.286, -72.902, -72.537, -72.311, -71.540, -71.775,
0492      . -71.572, -70.981, -70.813, -70.201, -69.489, -68.679, -68.434, -70.493
0493      data (Longitude(i),i= 3521, 3600),
0494      . -70.541, -70.547, -70.447, -70.716, -70.673, -70.768, -70.952, -71.053,
0495      . -71.158, -71.339, -71.492, -71.317, -71.393, -71.711, -71.654, -71.570,
0496      . -71.565, -71.565, -71.528, -71.446, -71.680, -71.644, -71.926, -72.042,
```

```
0497        . -72.199, -72.496, -72.712, -72.878, -73.145, -73.626, -73.659, -73.467,
0498        . -73.469, -73.263, -73.347, -73.662, -73.783, -73.911, -68.440, -68.771,
0499        . -69.382, -69.144, -69.106, -68.579, -68.715, -68.192, -67.729, -67.656,
0500        . -67.226, -66.678, -66.104, -66.389, -66.389, -65.775, -65.945, -66.677,
0501        . -67.206, -67.586, -67.374, -67.043, -66.450, -65.746, -65.451, -65.237,
0502        . -65.255, -65.028, -65.028, -64.431, -65.005, -64.435, -63.947, -63.595,
0503        . -64.215, -64.887, -65.028, -65.109, -65.108, -65.067, -64.412, -63.827/
0504        data (Longitude(i),i= 3601, 3680)/
0505        . -63.158, -62.547, -62.394, -62.332, -62.069, -62.327, -61.689, -61.043,
0506        . -60.399, -59.765, -59.143, -58.533, -57.952, -57.536, -57.170, -56.842,
0507        . -56.674, -57.117, -57.387, -57.166, -57.599, -58.137, -58.484, -58.472,
0508        . -58.459, -58.137, -58.200, -58.200, -58.046, -58.387, -58.425, -58.042,
0509        . -57.495, -56.920, -56.381, -55.772, -55.176, -54.572, -54.049, -53.671,
0510        . -53.374, -53.374, -52.924, -52.566, -52.382, -52.089, -52.094, -51.641,
0511        . -51.366, -51.092, -50.702, -51.154, -51.441, -51.852, -51.336, -50.924,
0512        . -50.583, -50.292, -50.094, -49.841, -49.505, -49.086, -48.715, -48.568,
0513        . -48.571, -48.629, -48.657, -48.556, -48.729, -48.188, -48.002, -38.920,
0514        . -38.954, -39.004, -39.031, -38.929, -39.009, -38.828, -38.324, -37.991,
0515        data (Longitude(i),i= 3681, 3760)/
0516        . -37.712, -37.510, -37.510, -37.344, -36.992, -36.604, -36.231, -35.891,
0517        . -35.553, -35.248, -35.150, -35.150, -34.978, -34.824, -34.812, -34.862,
0518        . -34.971, -35.104, -35.257, -35.581, -35.996, -35.996, -36.497, -36.981,
0519        . -37.409, -37.830, -38.196, -38.569, -38.978, -39.425, -39.859, -40.352,
0520        . -40.856, -41.084, -41.084, -41.589, -42.084, -42.564, -43.017, -43.517,
0521        . -4?.998, -44.414, -44.749, -44.653, -44.375, -44.521, -44.938, -45.338,
0522        . -45.839, -46.001, -48.003, -4?.?85, -47.206, -46.808, -46.388, -45.851,
0523        . -45.331, -44.835, -44.727, -44.?27, -44.415, -43.871, -43.340, -42.?95,
0524        . -42.249, -41.944, -41.515, -41.016, -41.062, -40.964, -40.964, -40.702,
0525        . -40.336, -40.140, -39.794, -39.716, -39.731, -39.584, -39.214, -39.216,
0526        data (Longitude(i),i= 3761, 3840)/
0527        . -39.134, -39.014, -38.920, -46.001, -46.497, -46.943, -47.419, -47.915,
0528        . -48.298, -48.436, -48.894, -49.280, -49.459, -49.692, -49.692, -49.479,
0529        . -49.867, -50.356, -50.712, -50.711, -50.826, -51.329, -51.669, -51.669,
0530        . -52.128, -52.633, -52.193, -51.771, -51.709, -51.709, -51.553, -51.255,
0531        . -50.863, -50.513, -50.181, -49.901, -50.136, -50.582, -50.772, -50.935,
0532        . -51.066, -51.166, -51.467, -51.684, -51.684, -51.766, -52.180, -52.589,
0533        . -52.958, -53.451, -53.919, -54.168, -54.168, -54.128, -54.624, -55.126,
0534        . -55.633, -56.121, -56.618, -57.107, -57.248, -57.249, -57.146, -57.162,
0535        . -60.426, -59.926, -59.506, -59.060, -58.683, -58.472, -58.612, -58.123,
0536        . -57.712, -57.296, -57.158, -71.325, -71.798, -71.806, -71.584, -71.827/
0537        data (Longitude(i),i= 3841, 3920)/
0538        . -72.117, -71.831, -71.489, -71.061, -71.091, -71.378, -71.527, -71.133,
0539        . -70.655, -70.168, -70.295, -70.206, -70.206, -69.806, -69.406, -68.895,
0540        . -68.455, -68.303, -67.866, -67.361, -66.854, -66.342, -65.918, -65.441,
0541        . -64.930, -64.448, -63.974, -64.289, -64.289, -63.780, -63.271, -62.764,
0542        . -62.256, -62.747, -62.801, -62.439, -62.439, -61.941, -61.444, -60.989,
0543        . -61.099, -61.106, -61.602, -61.129, -61.106, -61.602, -61.110, -60.604,
0544        . -60.425, -81.352, -81.352, -81.688, -81.688, -79.914, -79.914, -67.096,
0545        . -67.093, -67.496, -67.498, -67.352, -67.354, -67.055, -67.917, -67.066,
0546        . -67.056, -68.396, -68.154, -68.952, -69.831, -69.093, -68.391, -69.203,
0547        . -69.198, -70.348, -70.351, -70.993, -70.992, -72.028, -72.028, -71.282,
0548        data (Longitude(i),i= 3921, 4000)/
0549        . -71.281, -70.484, -70.580, -70.574, -70.483, -68.642, -69.514, -70.385,
0550        . -71.216, -72.001, -71.136, -70.279, -70.887, -70.079, -69.228, -69.378,
0551        . -69.376, -69.653, -70.182, -69.479, -70.305, -70.115, -69.624, -68.799,
0552        . -68.616, -71.671, -71.671, -72.396, -72.398, -72.294, -72.292, -73.179,
0553        . -73.180, -72.185, -72.809, -73.268, -72.446, -72.186, -73.688, -73.689,
0554        . -73.430, -73.420, -74.056, -74.057, -74.671, -73.959, -73.267, -74.027,
0555        . -74.711, -74.664, -73.398, -73.395, -73.351, -73.952, -74.136, -74.147,
0556        . -73.717, -73.724, -74.239, -74.238, -74.393, -74.402, -74.017, -74.023,
0557        . -74.775, -74.778, -75.023, -75.021, -75.109, -75.120, -75.314, -75.315,
0558        . -75.017, -75.017, -74.428, -74.432, -74.889, -74.889, -74.223, -74.222,
0559        data (Longitude(i),i= 4001, 4080)/
0560        . -74.715, -74.714, -75.321, -75.228, -75.317, -75.321, -74.781, -74.890,
```

```
0561      . -74.886, -74.497, -74.538, -74.784, -75.313,  75.312, -74.949, -74.958,
0562      . -75.523, -75.525, -74.885, -74.887, -75.274, -75.274, -75.634, -75.633,
0563      . -74.618, -74.762, -74.516, -74.623, -75.053, -75.057, -74.343, -74.342,
0564      . -75.599, -75.606, -75.242, -75.439, -75.196, -75.241, -75.247, -75.246,
0565      . -75.186, -75.181, -74.955, -74.959, -74.229, -74.230, -74.465, -74.464,
0566      . -75.063, -75.071, -74.700, -74.702, -74.486, -74.486, -73.931, -73.931,
0567      . -73.676, -73.677, -74.067, -74.069, -74.144, -74.142, -74.026, -74.030,
0568      . -73.790, -73.793, -74.388, -74.391, -74.333, -74.342, -74.306, -74.311,
0569      . -73.831, -73.844, -74.472, -74.473, -75.103, -75.100, -74.634, -74.635/
0570      data (Longitude(i),i= 4081, 4160)/
0571      . -74.803, -74.807, -74.285, -74.289, -73.771, -73.771, -73.303, -73.227,
0572      . -73.146, -73.301, -74.521, -74.523, -74.045, -74.049, -73.954, -73.967,
0573      . -74.399, -74.402, -74.039, -74.042, -73.867, -73.868, -73.993,  73.997,
0574      . -74.816, -74.816, -73.386, -73.392, -73.615, -73.628, -73.429, -73.438,
0575      . -73.894, -74.228, -74.202, -74.051, -73.440, -73.814, -73.520, -73.901,
0576      . -66.870, -66.864, -66.502, -66.499, -64.287, -64.287, -68.642, -68.644,
0577      . -68.638, -68.642, -68.615, -68.231, -68.000, -67.466, -66.764, -66.092,
0578      . -65.224, -65.965, -66.842, -67.714, -68.580, -68.638, -62.079, -62.079,
0579      . -48.567, -48.567, -48.591, -48.591, -50.831, -50.831, -50.674, -50.674,
0580      .  -51.126, -51.126, -50.564, -50.564, -50.360, -50.360, -50.069, -50.069/
0581      data (Longitude(i),i= 4161, 4240)/
0582      . -49.896, -50.300, -49.894, -49.896, -49.428, -49.428, -48.396, -48.531,
0583      . -48.784, -49.191, -49.627, -50.131, -50.621, -50.775, -50.783, -50.591,
0584      . -50.091, -49.599, -49.105, -48.606, -48.396, -50.033, -50.033, -50.368,
0585      . -50.368, -50.471, -50.471, -51.586, -51.455, -51.360, -51.729, -51.686,
0586      . -32.398, -32.398, -45.260, -45.260, -44.331, -44.331, -70.050, -70.050,
0587      . -69.146, -68.749, -69.163, -69.146, -68.377, -68.377, -66.616, -66.616,
0588      . -66.193, -66.193, -65.353, -65.353, -64.609, -64.609, -63.976, -63.976,
0589      . -64.331, -63.820, -64.323, -64.331, -61.079, -60.972, -61.405, -61.915,
0590      . -61.461, -61.464, -61.079, -60.525, -60.525, -61.610, -61.610, -61.163,
0591      . -61.163, -60.938, -60.938, -59.505, -59.505, -45.214, -45.214, -44.731/
0592      data (Longitude(i),i= 4241, 4320)/
0593      . -44.731, -37.039, -37.039, -36.087, -36.651, -37.404, -36.565, -36.059,
0594      . -36.087, -27.584, -27.584, -27.163, -27.163, -26.661, -26.661, -26.426,
0595      . -26.426, -26.259, -26.259, -26.592, -26.592, -27.344, -27.344, -29.844,
0596      . -29.844, -30.308, -30.308, -29.361, -29.361, -29.366, -29.366, -31.029,
0597      . -60.311, -59.773, -59.850, -60.517, -61.030, -59.720, -59.221, -58.705,
0598      . -57.906, -58.300, -59.039, -59.715, -61.106, -61.107,-109.406,-109.406,
0599      .-105.463,-105.463, -80.757, -80.757, -78.987, -78.987, -79.882, -79.882,
0600      . -80.082, -80.082, -90.523, -90.523, -89.658, -89.658, -89.362, -89.362,
0601      . -90.261, -90.261, -90.479, -90.479, -90.800, -90.800, -91.495, -91.284,
0602      . -91.473, -90.988, -90.959, -91.205, -91.495, -91.666, -91.666, -90.793/
0603      data (Longitude(i),i= 4321, 4400)/
0604      . -90.793, -69.524, -69.135, -68.864, -69.379, -69.765, -69.944, -69.523,
0605      .  75.820,  76.884,  77.905,  78.214,  79.201,  80.424,  81.462,  82.327,
0606      .  83.602,  84.893,  86.170,  87.387,  88.125,  88.917, 110.733, 111.802,
0607      . 112.978, 114.044, 114.711, 115.937, 116.846, 117.981, 119.272, 120.539,
0608      . 121.719, 121.973, 136.242, 137.485, 138.676, 139.890, 141.139, 142.331,
0609      . 143.607, 144.568, 144.637, 162.059, 163.417, 164.897, 166.396, 167.800,
0610      . 168.703, 169.906, 170.794, 169.874, 169.867, 169.109, 169.187, 162.796,
0611      . 163.065, 163.506, 164.419, 164.916, 166.907, 164.395, 162.640, 160.547,
0612      .  81.462,  82.240,  82.321, -75.980, -74.236, -72.592, -70.967, -69.257,
0613      . -67.888, -66.937, -66.963, -67.501, -67.509, -67.829, -68.423, -68.575,
0614      data (Longitude(i),i= 4401, 4480)/
0615      . -68.257, -65.806, -64.931, -64.167, -63.946, -63.168, -63.172, -62.495,
0616      . -61.428, -60.978, -59.894, -58.889, -57.951, -56.868, -57.924, -58.637,
0617      . -59.579, -60.533, -61.338, -62.095, -62.458, -61.228, -61.676, -62.972,
0618      . -63.561, -63.567, -64.144, -64.939, -65.593, -65.331, -65.487, -13.987,
0619      . -15.804, -16.331, -15.252, -14.202, -12.583, -11.505, -11.737, -10.611,
0620      . -10.106,  -8.901,  -8.631,  -7.621,  -6.803,  -6.107,  -5.999,  -5.995,
0621      .  -4.467,  -2.906,  -1.378,   0.194,   1.508,   2.853,   4.302,   5.804,
0622      .   7.170,   8.665,   9.989,   9.989,  11.097,  12.423,  13.748,  15.216,
0623      .  16.641,  17.998,  19.377,  20.734,  22.211,  23.716,  25.136,  26.538,
0624      .  27.979,  28.138,  28.141,  29.469,  30.914,  32.111,  33.011,  33.452,
```

110

```
0625        data (Longitude(i),i= 4481, 4560)
0626      .  34.817,  35.537,-167.004,-162.638,-156.730,-150.411,-146.292,-140.434,
0627      .-143.126,-148.133,-151.897,-153.221,-152.959,-152.298,-153.093,-154.883,
0628      .-155.337,-152.605,-149.649,-148.609,-145.584,-145.584,-110.090,-108.113,
0629      .-106.161,-104.316,-102.378,-100.491,-100.198,-101.489,-102.461,-100.796,
0630      . -99.180, -99.180,-100.672,-102.434,-103.541,-101.850,-100.124, -98.395,
0631      . -96.926, -95.188, -93.446, -91.714, -90.839, -90.830, -89.350, -87.641,
0632      . -85.967, -85.669, -83.929, -82.328, -81.301, -80.191, -78.870, -77.143,
0633      . -75.980, -68.255, -66.902, -67.366, -66.866, -66.470, -65.806, -65.488,
0634      . -64.770, -63.832, -63.566, -63.153, -62.517, -62.005, -62.222, -62.214,
0635      . -61.088, -61.712, -62.264, -60.799, -60.567, -59.848, -61.577, -60.689/
0636        data (Longitude(i),i= 4561, 4640)/
0637      . -61.552, -61.064, -61.061, -61.948,  63.373, -63.290, -64.838, -66.483,
0638      . -68.424, -69.988, -72.146, -74.282, -76.197, -77.430, -77.805, -77.803,
0639      . -78.477, -77.455, -76.273, -74.140, -73.654, -76.074, -78.424, -80.480,
0640      . -79.745, -77.643, -79.074, -81.564, -83.062, -83.281, -81.517, -80.576,
0641      . -80.580, -78.641, -76.169, -77.044, -79.892, -77.304, -75.182, -73.095,
0642      . -70.199, -67.469, -65.034, -62.399, -59.518, -59.100, -58.521, -58.526,
0643      . -56.409, -54.249, -51.063, -47.725, -44.187, -42.074, -39.365, -36.715,
0644      . -  .740, -31.114, -31.086, -28.503, -29.061, -28.658, -31.364, -34.056,
0645      . -36.099, -36.051, -34.892, -33.567, -31.653, -30.069, -28.578, -26.643,
0646      . -26.449, -26.446, -24.561, -22.515, -20.472, -18.483, -18.061, -17.223/
0647        data (Longitude(i),i= 4641, 4720)/
0648      . -15.865, -14.593, -13.981, -67.364, -66.935, -66.839, -67.597, -67.223,
0649      . -66.879,  69.690,  69.261,  67.989,  67.673,  69.145,  69.146,  68.237,
0650      .  67.790,  67.735,  69.328,  70.492,  71.309,  71.427,  72.306,  72.631,
0651      .  73.726,  74.985,  75.823,  88.917,  90.187,  91.404,  92.653,  93.915,
0652      .  95.170,  96.440,  97.706,  98.966, 100.091, 100.899, 102.071, 103.304,
0653      . 104.112, 104.119, 105.298, 106.453, 107.706, 108.688, 109.960, 110.721,
0654      . 110.733, 121.973, 123.171, 124.289, 125.524, 126.790, 127.243, 128.469,
0655      . 129.379, 130.116, 131.354, 132.588, 133.827, 134.113, 134.176, 135.182,
0656      . 135.029, 135.912, 136.242, 144.637, 145.477, 146.711, 148.049, 148.668,
0657      . 149.999, 151.012, 152.409, 153.732, 153.633, 154.559, 155.532, 156.950/
0658        data (Longitude(i),i= 4721, 4800)/
0659      . 158.318, 159.436, 160.454, 161.724, 162.059, 169.187, 167.438, 166.033,
0660      . 164.767, 165.333, 163.750, 161.625, 162.927, 162.796, 160.551, 160.467,
0661      . 159.109, 158.400, 161.200, 160.741, 161.997, 163.766, 165.240, 167.090,
0662      . 169.917, 172.596, 176.784,-179.982,-174.942,-174.102,  35.537,  36.289,
0663      .  37.561,  38.063,  39.449,  39.812,  40.536,  41.678,  42.745,  44.013,
0664      .  45.143,  46.292,  47.436,  48.607,  48.642,  49.918,  50.858,  50.860,
0665      .  50.169,  50.986,  51.903,  53.106,  54.338,  55.537,  56.301,  57.339,
0666      .  58.206,  59.130,  60.436,  61.684,  63.000,  64.322,  65.622,  66.938,
0667      .  68.257,  69.538,  69.775,  69.718,  69.692,-174.097,-169.169,-167.007,
0668      .-148.609,-147.243,-145.523,-146.363,-148.448,-150.827,-153.103,-155.216/
0669        data (Longitude(i),i= 4801, 4880)/
0670      .-153.866,-155.955,-158.103,-158.083,-158.083,-157.806,-155.593,-153.429,
0671      .-151.258,-149.662,-147.816,-145.982,-146.135,-148.164,-146.051,-146.051,
0672      .-144.243,-142.629,-142.631,-140.622,-139.349,-137.428,-135.882,-133.961,
0673      .-132.045,-130.142,-128.290,-127.006,-127.006,-125.113,-123.225,-121.318,
0674      .-119.461,-117.919,-116.167,-111.794,-113.188,-112.103,-111.547,-110.078,
0675      .-110.959,-110.089,-148.001,-148.001, 167.108, 169.286, 167.207, 166.682,
0676      . 167.108, 169.897, 169.897, 164.852, 164.852, 163.118, 163.118, 162.500,
0677      . 162.500,  92.403,  92.403, -90.808, -91.143, -91.619, -90.808, -74.221,
0678      . -74.351, -75.966, -75.386, -74.221, -73.502, -73.502, -67.742, -67.742,
0679      . -69.221, -68.654, -67.966, -67.673, -68.401, -69.221, -66.800, -66.800
0680        data (Longitude(i),i= 4881, 4960)
0681      . -66.162, -66.162, -63.542, -63.542, -64.254, -63.532, -63.590, -64.254,
0682      . -62.617, -62.617, -61.141, -60.118, -61.136, -61.141, -59.897, -59.897,
0683      . -59.674, -59.674, -59.200, -59.200, -58.976, -58.071, -58.944, -58.976,
0684      . -55.773, -55.343, -54.133, -54.133, -56.318, -56.318, -56.116, -55.526,
0685      . -56.413, -56.416, -56.000, -56.000, -57.956, -58.285, -57.167, -57.956,
0686      . -57.475, -57.475, -59.517, -59.517,  -5.996,  -5.996,  -2.719,  -2.719,
0687      .  -2.762,  -2.762,   2.923,   2.923,  13.145,  13.145,  16.134,  16.134,
0688      .-100.851, -99.187,-100.850,-100.851,-102.466,-101.030, -99.450, -97.916,
```

111

```
0689        . -96.566, -97.784, -99.443,-100.981,-102.466, -90.379, -90.379, -78.326,
0690        . -78.326, -79.191, -79.085, -77.559, -78.802, -79.191, -76.519, -75.341/
0691        data (Longitude(i),i= 4961, 5040)/
0692        . -74.006, -74.799, -76.299, -76.519, -75.587, -75.587, -72.677, -72.677,
0693        . -72.491, -71.087, -71.620, -73.237, -74.816, -75.142, -73.550, -73.066,
0694        . -71.497, -71.985, -71.392, -71.634, -72.166, -71.279, -69.994, -69.472,
0695        . -69.081, -68.503, -68.278, -68.164, -68.427, -69.171, -70.749, -70.817,
0696        . -72.491, -57.376, -57.376, -60.934, -60.934, -62.174, -62.360, -61.376,
0697        . -62.174, -61.076, -61.076, -60.722, -60.722, -71.765, -71.392, -69.576,
0698        . -67.391, -68.516, -69.791, -71.765, -62.441, -64.417, -67.031, -64.094,
0699        . -61.714, -60.334, -62.441, -52.938, -54.551, -52.426, -51.101, -50.677,
0700        . -50.258, -49.166, -47.303, -44.988, -43.899, -43.799, -43.360, -43.625,
0701        . -45.216, -48.047, -51.036, -52.938, -37.802, -40.346, -37.797, -37.802,
0702        data (Longitude(i),i= 5041, 5120)/
0703        . -20.294, -21.565, -20.073, -20.294,-105.176,-105.176, -60.665, -60.665,
0704        . -61.474, -61.474, -60.452, -60.452,  71.831,  71.831,  96.543,  96.543,
0705        . 100.496, 100.496, 103.318, 103.318,  48.321,  48.321,  47.408,  47.408,
0706        .-159.152,-161.832,-163.654,-163.031,-160.590,-159.009,-159.152,-149.220,
0707        .-149.220,-132.372,-132.372,-130.896,-130.896,-127.118,-127.118,-124.092,
0708        .-125.676,-127.163,-125.809,-124.441,-123.730,-124.092,-118.651,-120.269,
0709        .-122.076,-122.406,-120.630,-118.814,-118.651,-115.989,-115.989,  28.833,
0710        .  28.636,  28.583,  28.582,  28.110,  27.900,  27.485,  27.909,  28.013,
0711        .  28.013,  28.243,  28.840,  28.227,  27.572,  27.128,  26.587,  26.064,
0712        .  26.032,  26.032,  25.408,  24.749,  24.111,  24.297,  23.639,  23.000/
0713        data (Longitude(i),i= 5121, 5200)/
0714        .  22.721,  22.729,  22.729,  23.109,  23.011,  23.330,  23.884,  24.064,
0715        .  24.064,  23.475,  23.314,  22.899,  23.041,  22.492,  22.489,  22.489,
0716        .  22.263,  21.657,  21.618,  21.158,  21.637,  22.278,  22.847,  22.507,
0717        .  21.867,  21.230,  21.176,  21.176,  20.891,  20.549,  20.172,  20.009,
0718        .  20.010,  19.679,  19.400,  19.452,  19.444,  19.368,  19.369,  18.911,
0719        .  18.345,  17.805,  17.157,  16.807,  16.133,  15.599,  15.154,  14.893,
0720        .  14.682,  14.051,  13.570,  13.717,  13.717,  13.001,  12.314,  12.460,
0721        .  12.278,  12.573,  12.575,  13.132,  13.655,  13.870,  14.157,  14.661,
0722        .  15.228,  15.904,  15.988,  16.580,  17.197,  17.747,  17.954,  17.954,
0723        .  18.420,  18.354,  17.928,  17.319,  16.708,  16.522,  17.072,  17.139/
0724        data (Longitude(i),i= 5201, 5280)/
0725        .  16.548,  16.301,  16.000,  16.000,  15.865,  16.218,  16.041,  15.790,
0726        .  15.261,  14.909,  14.909,  14.488,  13.992,  13.448,  12.818,  12.286,
0727        .  11.800,  11.777,  11.777,  11.265,  10.792,  10.524,  10.277,   9.846,
0728        .   9.229,   8.527,   8.118,   7.529,   7.529,   6.951,   6.441,   5.756,
0729        .   5.123,   4.451,   3.761,   3.214,   3.053,   3.210,   3.210,   3.254,
0730        .   2.714,   2.176,   1.538,   0.926,   0.553,   0.449,   0.449,   0.043,
0731        .  -0.281,  -0.220,   0.065,  -0.466,  -0.733,  -0.721,  -0.721,  -1.351,
0732        .  -1.814,  -2.098,  -2.716,  -3.338,  -3.962,  -4.458,  -4.457,  -5.027,
0733        .  -5.465,  -6.074,  -6.398,  -6.707,  -7.321,  -7.414,  -7.414,  -7.993,
0734        .  -8.602,  -8.828,  -8.824,  -9.198,  -8.970,  -9.493,  -9.493,  -9.358/
0735        data (Longitude(i),i= 5281, 5360)/
0736        .  -9.086,  -8.787,  -8.765,  -8.668,  -8.819,  -8.750,  -8.750,  -8.886,
0737        .  -9.137,  -8.689,  -8.161,  -7.908,  -7.908,  -7.262,  -6.572,  -5.884,
0738        .  -5.218,  -4.541,  -4.483,  -4.483,  -3.808,  -3.143,  -2.451,  -1.769,
0739        .  -1.770,  -1.408,  -1.282,  -1.238,  -1.152,  -0.595,  -0.805,  -1.244,
0740        .  -1.244,  -1.192,  -1.773,  -2.035,  -2.010,  -2.010,  -2.539,  -3.183,
0741        .  -3.868,  -4.564,  -4.727,  -4.727,  -4.745,  -4.030,  -3.289,  -2.701,
0742        .  -2.314,  -2.315,  -1.560,  -1.572,  -1.841,  -1.181,  -0.411,  -0.218,
0743        .  -0.218,   0.484,   0.905,   1.521,   1.567,   2.183,   2.542,   2.542,
0744        .   3.233,   4.030,   4.040,   4.021,   4.021,   4.522,   4.713,   5.412,
0745        .   5.915,   4.104,   3.827,   4.104,   4.046,   4.168,   3.873,  10.903
0746        data (Longitude(i),i= 5361, 5440),
0747        .  10.479,   9.928,   8.805,   8.472,   8.133,   8.106,   8.137,  10.340,
0748        .  10.748,  10.023,   9.608,   9.621,   8.137,   8.957,   9.760,   8.847,
0749        .   8.586,   8.586,   9.496,   9.940,   9.940,  10.542,  10.542,   9.981,
0750        .   9.980,  10.339,   9.621,   9.749,   9.928,  10.610,  10.781,  10.758,
0751        .  12.443,  12.600,  11.714,  11.741,  12.557,  12.966,  13.647,  13.782,
0752        .   8.805,   8.815,   9.389,   8.543,   7.709,   7.016,   6.172,   5.915,
```

```
0753    .    8.959,    8.982,   14.226,   13.712,   13.712,   12.967,   12.175,   11.480,
0754    .   10.903,   19.624,   19.005,   19.004,   18.431,   17.566,   16.770,   16.103,
0755    .   15.271,   14.483,   14.226,   24.163,   25.211,   25.376,   24.524,   23.925,
0756    .   23.218,   22.453,   22.401,   22.401,   21.557,   21.078,   21.334,   21.538/
0757    data (Longitude(i),i= 5441, 5520)/
0758    .   21.450,   21.300,   21.300,   21.453,   22.408,   23.137,   23.225,   27.860,
0759    .   26.867,   25.858,   24.968,   24.015,   23.083,   23.226,   11.427,   11.248,
0760    .   11.794,   11.909,   12.236,   12.677,   12.557,   13.782,   14.237,   14.952,
0761    .   15.837,   16.211,   16.445,   16.649,   16.702,   16.701,   16.801,   16.194,
0762    .   17.139,   17.666,   18.602,   17.614,   16.703,   17.685,   18.646,   18.816,
0763    .   18.111,   17.232,   17.174,   17.274,   17.273,   17.403,   17.466,   17.954,
0764    .   18.591,   19.494,   20.033,   20.033,   20.852,   21.469,   21.045,   21.266,
0765    .   22.347,   23.430,   24.162,   30.784,   29.457,   30.733,   30.087,   28.571,
0766    .   28.197,   27.904,   26.773,   26.681,   25.817,   25.158,   25.281,   23.809,
0767    .   23.476,   22.261,   21.208,   21.209,   22.045,   20.592,   20.024,   20.382/
0768    data (Longitude(i),i= 5521, 5600)/
0769    .   18.995,   18.004,   17.470,   16.492,   17.838,   16.479,   16.252,   14.918,
0770    .   14.786,   14.786,   15.589,   14.337,   15.631,   14.351,   13.565,   13.216,
0771    .   12.607,   12.687,   11.800,   11.523,   10.512,   10.512,    9.976,   11.048,
0772    .   10.267,    9.162,    8.298,    7.201,    8.115,    7.054,    6.991,    5.977,
0773    .    5.086,    5.086,    5.921,    4.941,    5.220,    6.249,    7.292,    6.281,
0774    .    5.245,    5.261,    5.703,    6.459,    5.954,    5.521,    6.514,    6.034,
0775    .    5.457,    5.456,    6.073,    6.907,    7.852,    8.636,    9.278,   10.082,
0776    .   10.304,   11.007,   11.428,    5.376,    5.047,   25.657,   25.657,   24.655,
0777    .   24.655,   25.442,   25.442,   23.611,   23.611,   24.490,   24.490,   23.300,
0778    .   23.782,   24.214,   23.593,   23.183,   23.300,   24.374,   24.374,   24.441/
0779    data (Longitude(i),i= 5601, 5680)/
0780    .   24.441,   24.794,   24.794,   25.054,   25.054,   24.930,   24.930,   25.349,
0781    .   25.349,   25.572,   25.572,   25.308,   25.308,   25.384,   25.384,   26.321,
0782    .   25.834,   25.224,   24.634,   24.034,   23.570,   24.181,   24.761,   25.368,
0783    .   25.954,   26.321,   22.957,   22.957,   24.544,   24.544,   25.278,   25.278,
0784    .   24.653,   24.653,   23.501,   23.501,   23.547,   23.547,   20.706,   20.706,
0785    .   20.571,   20.571,   20.701,   20.701,   19.855,   19.855,   17.412,   17.412,
0786    .   16.861,   16.861,   16.721,   16.721,   16.388,   17.071,   16.384,   16.388,
0787    .   16.452,   16.452,   16.069,   16.069,   14.866,   14.866,   14.745,   15.245,
0788    .   14.735,   14.745,   14.358,   14.501,   14.301,   14.358,   14.594,   14.594,
0789    .   13.118,   13.118,   11.006,   11.006,    8.281,    8.281,    7.622,    7.622/
0790    data (Longitude(i),i= 5681, 5760)/
0791    .    7.339,    7.339,    6.797,    6.797,    6.328,    6.328,   12.055,   12.055,
0792    .   15.624,   15.280,   15.091,   15.120,   14.500,   14.023,   13.441,   12.911,
0793    .   12.485,   13.068,   13.676,   14.312,   14.909,   15.536,   15.624,   10.441,
0794    .   10.441,    9.256,    9.679,    9.711,    9.691,    9.605,    9.019,    8.419,
0795    .    8.383,    8.393,    8.376,    8.186,    8.849,    9.256,   14.324,   14.324,
0796    .   14.507,   14.507,  -25.193,  -25.193,  -25.843,  -25.241,  -25.843,  -27.380,
0797    .  -27.380,  -28.549,  -28.549,  -31.201,  -31.201,    4.224,    4.224,    3.186,
0798    .    3.295,    2.656,    3.093,    3.186,    1.558,    1.558,    1.449,    1.449,
0799    .   -0.748,   -0.748,   -7.871,   -7.871,    9.430,    9.527,    9.458,    9.281,
0800    .    8.672,    8.546,    8.960,    9.415,    9.430,   -1.229,   -1.229,   -1.299/
0801    data (Longitude(i),i= 5761, 5840)/
0802    .   -1.299,   -2.370,   -2.370,   -2.158,   -2.158,   -3.236,   -3.236,   -5.049,
0803    .   -5.049,   -2.082,   -2.082,   -2.500,   -2.500,    5.836,    5.836,    4.886,
0804    .    4.886,    5.580,    5.580,    5.674,    5.674,   -7.266,   -6.411,   -5.816,
0805    .   -5.455,   -6.029,   -6.077,   -6.103,   -9.628,   -9.329,  -10.007,   -9.607,
0806   -10.109,   -9.259,   -8.455,   -8.434,   -7.707,   -7.266,   -9.960,   -9.960,
0807    .   -3.979,   -3.191,   -2.253,   -2.066,   -2.438,   -3.144,   -3.325,   -3.325,
0808    .   -2.620,   -1.937,   -1.568,   -1.303,   -1.184,   -1.185,   -0.472,   -0.172,
0809    .   -0.260,   -0.261,    0.291,    0.215,    1.011,    1.714,    1.629,    1.157,
0810    .    0.487,    1.266,    1.442,    1.442,    0.971,    0.248,   -0.541,   -1.328,
0811    .   -2.079,   -2.870,   -3.530,   -4.305,   -5.010,   -5.072,   -4.474,   -3.759
0812    data (Longitude(i),i= 5841, 5920)
0813    .   -2.976,   -2.451,   -3.135,   -3.911,   -4.719,   -4.518,   -4.000,   -4.637,
0814    .   -4.760,   -4.760,   -4.218,   -3.421,   -2.898,   -2.790,   -3.559,   -3.298,
0815    .   -1.044,   -4.044,   -4.903,   -4.805,   -4.888,   -5.511,   -5.614,   -5.216,
0816    .   -5.122,   -5.122,   -5.909,   -5.789,   -5.830,   -5.194,   -5.263,   -5.263.
```

113

```
0817    .  -5.029,  -4.086,  -3.131,  -3.497,  -3.979,  -6.103,  -6.041,  -6.245,
0818    .  -6.401,  -6.402,  -7.152,  -7.873,  -8.541,  -9.303, -10.088, -10.339,
0819    . -10.338,  -9.771,  -9.369,  -9.628,  -6.244,  -6.383,  -6.272,  -6.244,
0820    .  -7.185,  -7.185,  -7.344,  -7.344,  -7.350,  -7.350,  -7.435,  -7.435,
0821    .  -6.323,  -5.905,  -6.773,  -6.323,  -6.326,  -6.326,  -6.487,  -6.487,
0822    .  -6.821,  -6.821,  -6.134,  -6.134,  -5.696,  -5.696,  -6.119,  -6.119/
0823       data (Longitude(i),i= 5921, 6000)/
0824    .  -6.117,  -6.117,   0.798,   0.798,  -5.072,  -4.775,  -4.474,  -3.042,
0825    .  -3.042,  -3.194,  -3.194,  -1.074,  -1.074,  -6.295,  -6.295,  -4.416,
0826    .  -4.416,  -4.353,  -4.353,  -5.266,  -5.266,  -8.552,  -8.552,  -0.795,
0827    .  -0.795,  -1.191,  -1.347,  -1.191,  -0.997,  -0.997,  11.231,  11.231,
0828    .  12.443,  12.120,  11.558,  11.099,  11.753,  11.741,  10.758,   9.908,
0829    .  10.614,  10.610,  10.557,  10.557,   9.076,   9.076,  14.776,  14.776,
0830    .  12.281,  12.281,  10.959,  10.959,   9.754,   9.754,  10.927,  10.927,
0831    .   8.928,   8.928,  25.552,  25.552,  23.749,  23.749,  23.156,  23.156,
0832    .  22.138,  23.340,  22.175,  22.138,  22.976,  22.976,  19.084,  19.084,
0833    .  18.191,  18.191,  20.468,  20.468,  19.932,  19.932,  19.902,  19.902/
0834       data (Longitude(i),i= 6001, 6080)/
0835    .  17.041,  17.344,  17.103,  17.041,  15.420,  15.420,  14.985,  15.538,
0836    .  15.046,  14.985,  14.811,  14.811,  14.211,  14.211,  13.489,  13.489,
0837    .  13.144,  13.144,  12.447,  12.447,  11.099,  11.099,   8.827,   8.827,
0838    .   9.144,   9.144,   8.184,   8.184,   6.039,   6.039,   5.803,   5.803,
0839    .   5.222,   5.222,   5.689,   5.689,   5.345,   5.345,   5.205,   5.205,
0840    .  11.737,  11.737,  17.110,  16.851,  16.570,  16.629,  17.010,  17.110,
0841    .  19.004,  18.927,  18.336,  18.110,  18.676,  19.004,  19.313,  19.313,
0842    .  19.225,  19.225,  24.586,  24.586,  21.418,  21.418,  22.954,  22.954,
0843    .  19.906,  19.906,  21.656,  21.656,  21.856,  21.856,  22.446,  22.446,
0844    . -14.559, -14.653, -13.805, -13.605, -14.417, -14.746, -14.746, -15.804/
0845       data (Longitude(i),i= 6081, 6160)/
0846    . -16.667, -17.777, -18.612, -19.709, -20.714, -21.365, -21.365, -22.502,
0847    . -21.705, -22.361, -23.454, -24.060, -24.058, -22.896, -21.755, -22.503,
0848    . -23.694, -23.677, -22.454, -23.139, -21.927, -21.377, -21.380, -21.446,
0849    . -20.325, -19.429, -18.239, -17.256, -16.270, -15.137, -14.558,  -6.717,
0850    .  -6.717,  -6.744,  -6.744,  -6.887,  -6.887,  -6.542,  -6.542,  -7.173,
0851    .  -7.173,  -7.409,  -7.409,  19.247,  19.894,  19.262,  19.247,   5.044,
0852    .   5.082,   5.871,   5.412,   5.376,  32.305,  31.882,  31.882,  31.315,
0853    .  30.788,  30.788,  30.228,  30.290,  30.290,  29.728,  29.225,  29.029,
0854    .  33.554,  33.850,  34.010,  34.255,  34.544,  34.778,  34.999,  35.278,
0855    .  35.661,  35.498,  35.663,  36.044,  36.446,  36.568,  29.028.  28.503/
0856       data (Longitude(i),i= 6161, 6240)/
0857    .  27.922,  27.374,  26.815,  26.241,  25.649,  25.152,  33.555,  33.422,
0858    .  33.047,  32.752,  32.752,  32.603,  32.410,  32.580,  25.152,  24.753,
0859    .  24.161,  23.609,  23.093,  22.643,  22.075,  21.488,  20.909,  20.403,
0860    .  20.005,  19.953,  20.143,  20.143,  19.954,  19.530,  18.966,  18.478,
0861    .  18.002,  17.461,  16.901,  16.315,  15.757,  15.429,  15.289,  14.724,
0862    .  14.205,  13.613,  13.017,  12.421,  11.889,  11.528,  11.528,  11.113,
0863    .  10.524,  10.059,  10.343,  10.784,  11.094,  11.098,  11.098,  10.820,
0864    .  10.456,  10.828,  11.072,  10.503,  10.208,   9.588,   8.999,   8.622,
0865    .   8.622,   8.004,   7.488,   7.487,   6.894,   6.289,   5.716,   5.107,
0866    .   4.527,   3.899,   3.293,   2.693,   2.308,   2.308,   1.692,   1.074/
0867       data (Longitude(i),i= 6241, 6320)/
0868    .   0.517,   0.063,  -0.529,  -1.117,  -1.558,  -2.139,  -2.199,  -2.199,
0869    .  -2.806,  -3.424,  -4.037,  -4.654,  -5.148,  -5.621,  -5.929,  -5.928,
0870    .  -6.114,  -6.318,  -6.590,  -6.917,  -7.430,  -7.971,  -8.519,  -8.881,
0871    .  -9.251,  -9.352,  -9.352,  -9.689,  -9.838,  -9.888,  -9.639,  -9.908,
0872    . -10.221, -10.616, -11.104, -11.485, -12.010, -12.558, -13.055, -13.176,
0873    . -15.678, -15.935, -16.197, -16.346, -16.714, -16.960, -17.029, -17.092,
0874    . -17.056, -13.176, -13.394, -13.578, -14.037, -14.489, -14.664, -14.845,
0875    . -14.896, -15.289, -15.678, -17.055, -16.553, -16.201, -16.288, -16.361,
0876    . -16.149, -16.051, -16.041, -16.172, -16.388, -16.528, -16.528, -16.528,
0877    . -16.661, -16.949, -17.336, -17.005, -16.782, -16.548, -16.037, -15.525
0878       data (Longitude(i),i= 6321, 6400)/
0879    . -16.027, -16.529, -16.783, -16.364, -15.851, -15.390, -15.815, -16.328,
0880    . -16.720, -16.720, -16.206, -15.779, -15.272, -15.477, -15.254, -14.746,
```

```
0881      . -14.664, -14.276, -13.852, -13.518, -13.306, -13.302, -13.298, -13.181,
0882      . -12.924, -12.642, -12.306, -11.833, -11.419, -11.012, -10.640, -10.199,
0883      .  -9.816,  -9.456,  -9.053,  -8.604,  -8.148,  -7.682,  -7.526,  -7.526,
0884      .  -7.057,  -6.596,  -6.124,  -5.639,  -5.134,  -4.630,  -4.129,  -4.634,
0885      .  -4.137,  -3.652,  -3.182,  -3.104,  -3.104,  -2.611,  -2.147,  -1.676,
0886      .  -1.198,  -0.721,  -0.271,   0.185,   0.685,   1.115,   1.199,   1.199,
0887      .   1.688,   2.191,   2.695,   3.202,   3.672,   4.141,   4.617,   4.981,
0888      .   5.432,   5.492,   5.583,   5.995,   6.500,   6.753,   7.160,   7.664/
0889          data (Longitude(i),i= 6401, 6480)/
0890      .   7.764,   7.764,   8.270,   8.775,   8.988,   8.988,   9.385,   9.810,
0891      .   9.942,   9.844,   9.815,   9.815,   9.752,   9.486,   9.849,   9.804,
0892      .   9.804,   9.622,   9.985,   9.492,   9.298,   9.113,   9.113,   8.982,
0893      .   9.288,   9.560,   9.587,  10.045,  10.292,  10.642,  10.981,  11.15,
0894      .   9.557,   9.107,   9.317,   9.580,   9.587,  11.160,  11.512,  11.811,
0895      .  12.011,  13.182,  12.685,  12.216,  12.162,  12.011,  12.989,  12.735,
0896      .  12.518,  12.353,  12.282,  12.106,  11.997,  11.787,  11.819,  11.762,
0897      .  11.755,  13.182,  12.695,  12.446,  12.736,  12.912,  13.108,  13.322,
0898      .  13.367,  12.990,  13.145,  13.145,  13.284,  13.504,  13.773,  13.829,
0899      .  13.786,  13.657,  13.364,  12.988,  11.754,  11.732,  11.898,  12.179/
0900          data (Longitude(i),i= 6481, 6560)/
0901      .  12.490,  12.737,  12.989,  13.235,  13.392,  13.685,  13.952,  14.267,
0902      .  14.509,  14.529,  14.528,  14.448,  14.449,  14.500,  14.469,  14.642,
0903      .  14.858,  14.874,  14.982,  15.156,  15.241,  15.445,  15.705,  16.106,
0904      .  16.499,  16.488,  16.739,  16.980,  17.148,  17.336,  17.593,  17.885,
0905      .  18.221,  18.340,  18.188,  18.079,  18.400,  18.313,  18.492,  18.491,
0906      .  19.101,  19.529,  20.137,  20.662,  21.269,  21.871,  22.368,  22.974,
0907      .  23.567,  24.170,  24.756,  25.314,  25.705,  25.706,  26.204,  26.796,
0908      .  27.316,  27.804,  28.251,  28.695,  29.098,  29.471,  29.934,  30.298,
0909      .  30.600,  30.864,  31.063,  31.063,  31.300,  31.680,  32.146,  32.454,
0910      .  32.602,  32.746,  32.884,  32.893,  32.892,  32.927,  32.546,  32.858/
0911          data (Longitude(i),i= 6561, 6640)/
0912      .  33.304,  33.822,  34.343,  34.484,  34.485,  34.997,  35.369,  35.534,
0913      .  35.480,  35.504,  35.540,  35.296,  35.118,  35.121,  35.121,  34.777,
0914      .  34.754,  34.590,  35.107,  35.489,  35.852,  36.253,  40.738,  40.647,
0915      .  40.599,  40.591,  40.477,  40.549,  40.487,  40.457,  40.547,  40.438,
0916      .  36.253,  36.590,  36.949,  37.280,  37.747,  38.224,  38.723,  39.193,
0917      .  39.645,  40.007,  40.299,  40.622,  40.642,  40.738,  40.438,  40.056,
0918      .  39.740,  39.632,  39.445,  39.286,  39.446,  39.350,  39.336,  38.989,
0919      .  38.787,  38.974,  39.096,  39.205,  39.204,  39.570,  39.809,  39.961,
0920      .  40.166,  40.486,  40.885,  41.315,  41.559,  41.557,  41.836,  42.105,
0921      .  42.449,  42.796,  42.894,  42.891,  43.234,  43.579,  43.967,  44.336/
0922          data (Longitude(i),i= 6641, 6720)/
0923      .  44.743,  45.173,  45.611,  46.040,  46.392,  46.756,  47.096,  47.453,
0924      .  47.491,  47.491,  47.819,  48.111,  48.385,  48.681,  48.944,  49.095,
0925      .  49.311,  49.588,  49.587,  49.819,  50.102,  50.326,  50.610,  50.817,
0926      .  50.904,  51.182,  51.413,  51.413,  51.122,  51.096,  51.150,  50.640,
0927      .  50.278,  49.785,  49.287,  48.925,  48.924,  48.416,  47.930,  47.411,
0928      .  46.968,  46.506,  45.999,  45.490,  45.033,  44.524,  44.055,  43.678,
0929      .  43.401,  43.101,  43.120,  43.379,  43.105,  42.681,  43.101,  38.612,
0930      .  38.831,  39.020,  39.161,  43.121,  42.729,  42.347,  42.023,  41.612,
0931      .  41.310,  41.252,  41.252,  40.763,  40.290,  40.057,  39.566,  39.288,
0932      .  39.188,  39.159,  38.611,  38.137,  37.792,  37.412,  37.280,  37.198/
0933          data (Longitude(i),i= 6721, 6800)/
0934      .  37.230,  37.122,  36.977,  36.841,  36.565,  33.473,  33.472,  33.256,
0935      .  33.256,  11.279,  11.279,  11.057,  11.057,   8.947,   8.947,  -3.027,
0936      .  -3.027, -13.455, -13.455, -14.088, -14.423, -13.902, -14.088, -15.818,
0937      . -15.818, -16.236, -16.758, -16.202, -16.236, -17.269, -17.269, -18.057,
0938      . -18.057, -17.783, -17.783, -16.020, -16.020, -16.997, -16.997, -16.371,
0939      . -16.371, -24.701, -24.701, -24.468, -24.468, -23.728, -23.728, -23.216,
0940      . -23.216, -22.909, -22.909, -22.991, -22.991, -24.425, -24.425, -25.046,
0941      . -25.046, -25.317, -25.317, -16.336, -16.336, -15.985, -15.985, -15.471,
0942      . -15.471, -15.942, -15.942, -16.239, -16.239, -16.151, -16.151, -16.059,
0943      . -16.059, -15.981, -15.981, -15.946, -15.946, -15.661, -15.661, -12.885/
0944          data (Longitude(i),i= 6801, 6880)/
```

115

```
0945    . -12.885,    5.634,    5.634,    6.683,    6.683,    7.424,    7.424,    8.919,
0946    .   8.744,    8.902,    8.919,   11.754,   11.754,   32.972,   32.972,   35.476,
0947    .  35.476,   43.290,   43.478,   43.274,   43.290,   43.652,   43.652,   44.207,
0948    .  44.207,   45.097,   45.097,   45.283,   45.283,   46.280,   46.280,   46.448,
0949    .  46.448,   39.722,   39.722,   39.189,   39.428,   39.356,   39.189,   39.696,
0950    .  39.678,   39.767,   39.696,   39.669,   39.669,   41.003,   41.003,   39.978,
0951    .  39.978,   55.710,   55.710,   55.713,   55.713    55.464,   55.464,   56.284,
0952    .  56.284,   56.532,   56.532,   56.564,   56.564,   57.750,   57.333,   57.750,
0953    .  63.478,   63.478,   48.217,   48.217,   42.741,   42.741,   45.614,   45.051,
0954    .  44.542,   44.073,   43.852,   43.663,   43.746,   43.597,   43.319,   43.287/
0955    data (Longitude(i),i= 6881,  6960)/
0956    .  43.421,   43.758,   43.946,   44.250,   44.251,   44.251,   44.481,   44.465,
0957    .  44.231,   44.089,   44.004,   43.937,   44.209,   44.416,   44.753,   45.209,
0958    .  45.699,   46.223,   46.649,   47.171,   47.261,   47.286,   48.433,   48.301,
0959    .  48.116,   47.915,   47.816,   47.674,   47.537,   47.322,   47.127,   46.689,
0960    .  46.131,   45.653,   45.614,   47.287,   47.754,   47.956,   47.890,   48.403,
0961    .  48.816,   48.940,   49.039,   49.274,   49.274,   49.544,   49.800,   49.976,
0962    .  50.140,   50.206,   50.239,   50.486,   50.368,   50.166,   50.166,   49.903,
0963    .  49.688,   49.861,   49.590,   49.447,   49.460,   49.365,   49.171,   48.999,
0964    .  48.834,   48.654,   48.502,   48.433,   51.759,   51.759,   50.223,   50.223,
0965    .  37.654,   37.654,   -9.892,   -9.892,  -12.210,  -12.210,   -5.715,   -5.715/
0966    data (Longitude(i),i= 6961,  7040)/
0967    . -14.392,  -14.392,   30.532,   30.502,   30.869,   31.267,   31.241,   30.840,
0968    .  30.531,   29.046,   29.122,   29.287,   29.369,   29.570,   29.596,   29.806,
0969    .  29.920,   29.945,   29.945,   29.861,   30.317,   30.561,   30.616,   30.884,
0970    .  31.146,   30.644,   30.463,   30.299,   30.299,   30.185,   29.805,   29.492,
0971    .  29.320,   29.340,   29.149,   29.100,   29.237,   29.046,   27.022,   27.326,
0972    .  27.569,   27.968,   28.430,   28.954,   28.482,   28.051,   27.614,   27.249,
0973    .  27.022,   33.926,   34.071,   34.201,   34.215,   34.325,   34.054,   34.179,
0974    .  34.334,   34.424,   34.544,   35.048,   34.857,   34.809,   34.821,   34.708,
0975    .  34.959,   34.963,   34.963,   34.663,   34.657,   34.553,   34.308,   33.928,
0976    .  31.613,   31.752,   31.848,   31.792,   31.976,   32.254,   32.254,   32.735/
0977    data (Longitude(i),i= 7041,  7120)/
0978    .  33.213,   33.714,   34.022,   34.186,   34.186,   34.678,   34.425,   34.131,
0979    .  33.816,   33.566,   33.203,   33.704,   33.432,   33.426,   33.426,   32.952,
0980    .  32.832,   32.665,   32.161,   31.844,   31.631,   31.614,   37.325,   37.001,
0981    .  37.438,   37.451,   37.325,   49.827,   49.827,   35.922,   35.911,   36.068,
0982    .  35.590,   34.987,   34.370,   33.975,   33.368,   32.806,   32.805,   32.277,
0983    .  31.790,   31.228,   30.601,   30.523,   29.915,   29.295,   29.014,   28.622,
0984    .  28.622,   28.019,   27.406,   28.012,   27.449,   27.222,   27.244,   26.674,
0985    .  26.349,   26.894,   27.015,   26.628,   26.074,   26.073,   26.168,   26.638,
0986    .  27.269,   27.927,   28.564,   29.123,   29.778,   29.158,   29.734,   30.281,
0987    .  30.281,   30.921,   31.487,   32.073,   32.656,   33.293,   33.968,   34.641/
0988    data (Longitude(i),i= 7121,  7200)/
0989    .  35.021,   35.021,   35.418,   36.087,   36.462,   37.117,   37.780,   38.390,
0990    .  39.036,   39.701,   39.997,   35.896,   35.821,   35.922,   35.896,   35.973,
0991    .  35.973,   35.649,   35.448,   35.213,   35.100,   35.100,   34.914,   34.782,
0992    .  34.547,   34.490,   34.490,   34.219,   34.219,   33.684,   33.102,   32.769,
0993    .  32.770,   32.306,   32.580,   32.721,   33.071,   33.220,   33.564,   33.564,
0994    .  33.931,   34.448,   34.516,   34.516,   34.677,   34.891,   34.961,   34.961,
0995    .  34.979,   34.978,   34.961,   34.961,   34.841,   34.734,   34.708,   34.708,
0996    .  35.233,   35.233,   35.518,   35.822,   36.163,   36.436,   36.695,   37.075,
0997    .  37.279,   37.424,   37.843,   37.843,   38.298,   38.636,   38.789,   39.063,
0998    .  39.037,   39.088,   39.191,   39.504,   39.569,   39.568,   40.006,   40.456,
0999    data (Longitude(i),i= 7201,  7280)/
1000    .  40.797,   41.049,   41.213,   41.509,   41.784,   42.203,   42.357,   42.716,
1001    .  42.821,   42.696,   42.795,   42.955,   43.076,   43.255,   43.228,   43.229,
1002    .  43.476,   43.931,   44.436,   44.934,   45.393,   45.799,   46.313,   46.823,
1003    .  47.314,   47.742,   48.253,   48.759,   49.073,   49.183,   49.182,   49.639,
1004    .  50.149,   50.622,   51.130,   51.623,   52.076,   52.195,   52.543,   53.031,
1005    .  53.542,   54.010,   54.531,   55.059,   55.204,   55.204,   55.331,   55.795,
1006    .  56.320,   56.613,   56.884,   57.393,   57.747,   57.741,   57.813,   57.813,
1007    .  58.037,   59.476,   50.826,   50.546,   50.239,   49.993,   50.007,   49.686,
1008    .  49.304,   48.834,   48.632,   48.420,   48.417,   58.476,   58.707,   59.034,
```

```
1009        .  59.429,  59.681,  59.811,  59.336,  59.007,  58.674,  58.143,  57.601/
1010        data (Longitude(i),i= 7281, 7360)/
1011        .  57.090,  56.734,  56.454,  56.383,  56.270,  56.271,  56.371,  56.080,
1012        .  56.081,  55.745,  55.331,  54.938,  54.606,  54.108,  53.630,  53.080,
1013        .  52.523,  51.994,  51.616,  51.219,  51.218,  51.592,  51.489,  51.388,
1014        .  50.899,  50.758,  50.826,  48.418,  48.164,  47.802,  47.982,  47.944,
1015        .  47.944,  48.517,  48.537,  48.538,  48.877,  49.405,  49.986,  50.293,
1016        .  50.669   50.900,  51.107,  51.264,  34.582,  34.097,  33.716,  33.207
1017        .  32.598,  32.549,  33.122,  33.736,  34.311,  34.582,  26.011,  26.011,
1018        .  26.420,  26.420,  26.161,  26.161,  26.361,  26.361,  27.065,  27.065,
1019        .  27.038,  27.038,  27.355,  27.355,  27.863,  27.863,  28.207,  27.916,
1020        .  28.207,  27.231,  27.231,  41.861,  41.861,  41.754,  42.178,  41.754/
1021        data (Longitude(i),i= 7361, 7440)/
1022        .  52.092,  52.092,  54.221,  53.832,  53.353,  53.854,  54.221,  56.052,
1023        .  56.052,  58.902,  58.728,  58.900,  58.902,  53.862,  53.862,  54.479,
1024        .  54.479,  54.467,  54.467,  53.335,  53.335,  50.502,  50.502,  48.186,
1025        .  48.187,  47.149,  47.149,  47.132,  47.132,  46.814,  46.814,  46.750,
1026        .  46.751,  46.905,  46.905,  46.898,  46.898,  32.299,  32.299,  32.385,
1027        .  32.387,  47.630,  47.051,  47.632,  47.630,  32.307,  32.319,  32.319,
1028        .  32.300,  32.299,  32.355,  32.568,  32.580,  77.567,  77.567,  77.534,
1029        .  77.534,  70.078,  69.344,  68.765,  68.919,  69.587,  70.349,  69.803,
1030        .  70.078,  69.326,  69.326,  73.234,  73.234,   3.354,   3.354, 172.661,
1031        . 172.661, 173.469, 173.469, 177.639, 177.639, 178.694, 179.398, 178.661/
1032        data (Longitude(i),i= 7441, 7520)/
1033        . 178.694,-178.784,-178.784,-177.925,-177.925,-177.046,-177.046,-176.549,
1034        .-176.549,-176.276,-176.276,-176.160,-176.160,-176.032,-176.032,-176.042,
1035        .-176.042,-175.865,-175.865,-175.718,-175.718,-174.159,-174.706,-174.159,
1036        .-174.011,-173.191,-174.011,-174.011,-172.407,-172.407,-171.250,-171.250,
1037        .-170.602,-170.602,-170.104,-170.104,-169.693,-169.693,-169.678,-169.678,
1038        .-167.795,-168.369,-169.042,-168.625,-167.896,-167.795,-166.258,-166.653,
1039        .-167.423,-166.816,-166.258,-165.659,-165.659,-165.268,-165.268,-164.935,
1040        .-164.935,-165.514,-165.514, 190.274, 190.274, 189.857, 189.857, 193.891,
1041        . 194.386, 193.418, 192.619, 193.530, 193.891, 187.086, 187.086, 190.334,
1042        . 189.621, 188.506, 189.340, 190.332, 190.334, 190.923, 190.923, 179.663/
1043        data (Longitude(i),i= 7521, 7600)/
1044        . 179.663,-155.823,-155.336,-154.971,-155.276,-155.636,-155.886,-155.910,
1045        .-155.823,-156.270,-156.270,-156.568,-156.568,-156.914,-156.914,-157.175,
1046        .-157.175,-157.967,-157.967,-159.318,-159.318,-160.057,-160.057,-160.534,
1047        .-160.534,-161.938,-161.938,-164.697,-164.697,-167.995,-167.995,-171.721,
1048        .-171.721,-173.947,-173.947,-177.322,-177.322,-177.358,-177.358,-178.294,
1049        .-178.294,-169.543,-169.543,  28.833,  29.489,  29.661, 351.674, 351.674,
1050        .  18.984,  18.984,  25.414,  25.414,  23.075,  23.928,  21.619,  21.284,
1051        .  23.075,  21.996,  21.996,  26.614,  26.614,  28.654,  28.654,  24.852,
1052        .  27.137,  25.845,  24.147,  21.606,  19.481,  17.880,  20.329,  22.214,
1053        .  24.747,  24.852,  33.095,  33.095,  10.679,  11.765,  12.248,  13.799/
1054        data (Longitude(i),i= 7601, 7680)/
1055        .  14.497,  16.591,  14.643,  16.883,  14.849,  14.406,  16.499,  17.625,
1056        .  18.373,  19.045,  21.203,  19.822,  18.686,  17.008,  15.886,  16.369,
1057        .  15.359,  12.632,  10.679,  19.247,  19.894,  19.262,  19.247,  56.808,
1058        .  57.069,  57.184,  57.429,  57.980,  58.529,  59.038,  59.594,  60.144,
1059        .  60.696,  61.232,  61.610,  61.610,  62.163,  62.718,  63.270,  63.804,
1060        .  64.354,  64.906,  65.458,  66.011,  66.438,  66.438,  66.739,  67.070,
1061        .  67.359,  67.532,  67.531,  67.647,  67.647,  67.857,  67.857,  68.011,
1062        .  68.011,  68.196,  68.196,  68.741,  68.433,  68.806,  69.289,  69.834,
1063        .  70.371,  70.023,  69.513,  69.211,  69.211,  69.403,  69.803,  70.142,
1064        .  70.582,  71.117,  71.606,  71.928,  71.929,  72.258,  72.099,  72.554,
1065        data (Longitude(i),i= 7681, 7760)/
1066        .  72.556,  73.101,  72.631,  72.846,  72.885,  72.704,  72.716,  73.047,
1067        .  73.046,  72.851,  73.014,  73.104,  73.234,  73.306,  73.386,  73.517,
1068        .  73.890,  74.030,  74.304,  74.511,  74.509,  74.631,  74.731,  74.855,
1069        .  75.074,  75.397,  75.617,  75.844,  76.021,  76.194,  76.417,  76.395,
1070        .  76.296,  76.395,  76.416,  76.296,  76.431,  76.693,  77.015,  77.446,
1071        .  77.914,  78.112,  78.112,  78.384,  78.868,  79.373,  78.925,  79.143,
1072        .  79.236,  79.236,  79.733,  79.845,  79.839,  79.786,  79.988,  80.225,
```

117

```
1073    .   80.314,    80.057,    80.141,    80.147,    80.147,    80.049,    80.209,    80.592,
1074    .   81.108,    81.384,    81.908,    82.334,    82.446,    82.906,    83.334,    83.690,
1075    .   84.124,    84.123,    84.474,    84.732,    85.153,    85.427,    85.954,    86.435/
1076        data (Longitude(i),i= 7761,  7840)/
1077    .   86.808,    86.921,    86.981,    87.471,    87.961,    87.933,    88.079,    88.077,
1078    .   88.204,    88.624,    88.707,    89.037,    89.037,    89.462,    89.545,    89.870,
1079    .   89.984,    90.044,    89.915,    90.454,    90.516,    90.538,    90.252,    90.252,
1080    .   90.629,    90.879,    91.411,    91.757,    91.908,    91.988,    92.209,    92.261,
1081    .   92.261,    92.507,    92.793,    93.159,    93.687,    93.920,    93.676,    94.058,
1082    .   94.068,    93.910,    93.871,    94.068,    94.039,    94.308,    94.453,    94.569,
1083    .   94.457,    94.270,    94.211,    94.613,    94.792,    94.792,    95.144,    95.646,
1084    .   96.031,    96.186,    96.691,    96.904,    96.881,    96.881,    97.259,    97.342,
1085    .   97.629,    97.728,    97.820,    97.993,    98.053,    98.096,    98.468,    98.471,
1086    .   98.470,    98.637,    98.704,    98.654,    98.718,    98.576,    98.525,    98.744/
1087        data (Longitude(i),i= 7841,  7920)/
1088    .   98.743,    98.569,    98.406,    98.287,    98.233,    98.704,    99.060,    99.348,
1089    .   99.734,    99.930,   100.128,   102.096,   101.719,   101.354,   100.848,   100.400,
1090    .  100.151,   100.480,   100.382,   100.279,    99.957,    99.890,    99.386,    99.377,
1091    .   99.377,    99.174,    99.174,    99.309,    99.491,    99.675,    99.904,    99.984,
1092    .  100.084,   100.017,    99.997,   100.497,   100.936,   100.847,   101.246,   101.246,
1093    .  101.757,   102.156,   102.548,   102.909,   102.912,   100.127,   100.354,   100.384,
1094    .  100.422,   100.624,   100.769,   101.044,   101.310,   101.477,   101.877,   102.316,
1095    .  102.559,   102.559,   102.958,   103.401,   103.899,   104.121,   103.922,   103.582,
1096    .  103.454,   103.450,   103.450,   103.421,   103.460,   103.332,   103.080,   102.669,
1097    .  102.373,   102.096,   102.912,   103.103,   103.605,   103.546,   104.054,   104.446/
1098        data (Longitude(i),i= 7921,  8000)/
1099    .  104.446,   104.859,   104.851,   104.816,   104.814,   104.963,   105.401,   105.825,
1100    .  106.164,   106.164,   105.955,   106.361,   106.273,   106.781,   106.577,   106.577,
1101    .  107.088,   107.592,   108.043,   108.484,   108.925,   109.233,   109.212,   109.272,
1102    .  109.325,   109.325,   109.244,   109.232,   109.078,   108.921,   108.691,   108.387,
1103    .  108.061,   107.611,   107.232,   107.117,   107.117,   106.738,   106.396,   106.239,
1104    .  105.848,   105.607,   105.805,   105.974,   106.431,   106.500,   106.500,   106.788,
1105    .  107.240,   107.632,   107.988,   114.033,   114.220,   107.988,   108.522,   109.059,
1106    .  109.586,   109.950,   109.951,   109.691,   109.820,   110.269,   110.316,   110.394,
1107    .  110.909,   111.185,   111.185,   111.707,   112.192,   112.730,   113.241,   113.165,
1108    .  113.542,   113.492,   113.686,   113.796,   113.959,   113.960,   114.034,   114.221/
1109        data (Longitude(i),i= 8001,  8080)/
1110    .  114.501,   114.499,   114.933,   115.470,   115.994,   116.507,   116.821,   117.265,
1111    .  117.701,   117.788,   117.792,   118.327,   118.782,   119.214,   119.574,   119.235,
1112    .  119.671,   119.573,   120.106,   120.252,   120.666,   120.849,   120.561,   120.558,
1113    .  121.112,   121.512,   121.618,   121.910,   121.662,   121.221,   120.680,   120.146,
.114     .  120.146,   120.685,   121.185,   121.719,   121.588,   121.131,   120.676,   120.090,
1115    .  119.674,   120.121,   120.693,   121.227,   121.785,   121.467,   121.100,   120.887,
1116    .  120.640,   120.411,   120.144,   119.600,   124.368,   124.628,   125.268,   125.249,
1117    .  125.642,   125.000,   124.722,   125.279,   125.909,   126.540,   126.697,   128.366,
1118    .  127.894,   127.375,   127.531,   128.079,   128.642,   129.173,   129.729,   129.802,
1119    .  129.882,   130.354,   130.696,   126.694,   126.707,   126.867,   126.243,   126.548/
1120        data (Longitude(i),i= 8081,  8160)/
1121    .  126.867,   126.871,   126.506,   126.383,   126.548,   127.156,   127.755,   128.329,
1122    .  128.906,   128.904,   128.903,   129.357,   129.553,   129.438,   129.419,   129.219,
1123    .  128.889,   128.566,   128.366,   130.694,   131.233,   131.568,   132.211,   132.678,
1124    .  133.353,   133.994,   134.554,   135.134,   135.462,   135.848,   136.313,   136.785,
1125    .  137.268,   137.731,   138.121,   138.402,   138.781,   139.163,   139.606,   140.097,
1126    .  140.171,   140.169,   140.339,   140.548,   140.635,   140.439,   140.670,   140.902,
1127    .  141.365,   141.143,   141.063,   141.396,   141.396,   140.742,   140.259,   139.508,
1128    .  138.653,   138.678,   137.912,   137.471,   136.703,   136.768,   136.078,   135.241,
1129    .  135.755,   136.348,   137.083,   137.734,   138.149,   138.712,   139.429,   140.028,
1130    .  140.623,   141.239,   141.881,   142.555,   143.512,   144.503,   145.492,   146.472/
1131        data (Longitude(i),i= 8161,  8240)/
1132    .  147.403,   148.362,   149.179,   150.175,   151.164,   151.941,   151.278,   152.238,
1133    .  153.206,   153.741,   153.743,   154.694,   154.087,   154.499,   155.077,   155.888,
1134    .  156.376,   156.950,   157.936,   158.966,   160.034,   159.925,   160.237,   161.201,
1135    .  161.897,   162.772,   163.081,   163.707,   164.793,   165.634,   165.648,   164.556,
1136    .  164.076,   163.803,   163.501,   162.789,   161.914,   161.404,   160.770,   160.079,
```

```
1137        . 159.614, 159.006, 158.304, 157.489, 156.966, 156.476, 155.923, 155.713,
1138        . 155.595, 155.579, 155.683, 155.858, 155.971, 156.086, 156.241, 156.451,
1139        . 156.492, 156.701, 157.426, 157.990, 158.395, 158.419, 158.424, 158.953,
1140        . 159.732, 159.840, 160.121, 160.783, 161.644, 162.022, 161.718, 161.982,
1141        . 162.509, 163.218, 162.815, 163.117, 162.281, 162.050, 162.495, 163.008/
1142        data (Longitude(i),i= 8241, 8320)/
1143        . 163.390, 163.360, 163.360, 164.323, 165.288, 166.099, 166.118, 166.912,
1144        . 167.747, 168.766, 169.711, 170.189, 170.618, 171.408, 172.151, 172.806,
1145        . 173.736, 174.725, 175.600, 176.528, 176.987, 176.976, 177.838, 178.848,
1146        . 179.501, 179.265, 178.356, 178.395, 177.552, 176.597, 175.429, 176.592,
1147        . 177.728, 178.909, 179.822, 180.006, 180.630, 180.184, 180.816, 181.132,
1148        . 181.672, 182.859, 183.997, 184.240, 185.303, 186.143, 186.589, 186.588,
1149        . 187.662, 187.022, 187.667, 188.720, 189.507, 189.357, 188.529, 187.376,
1150        . 186.109, 185.749, 185.070, 184.745, 183.860, 182.756, 181.736, 180.616,
1151        . 179.992, 178.879, 177.620, 176.331, 174.889, 173.737, 173.761, 172.347,
1152        . 170.920, 170.166, 170.931, 170.623, 169.417, 168.230, 167.816, 166.569/
1153        data (Longitude(i),i= 8321, 8400)/
1154        . 165.137, 163.705, 162.271, 161.422, 161.292, 161.001, 160.321, 159.689,
1155        . 159.733, 158.557, 157.046, 155.537, 154.014, 152.542, 151.652, 151.661,
1156        . 150.725, 149.389, 149.441, 147.863, 146.563, 145.390, 145.732, 144.114,
1157        . 145.754, 144.380, 142.710, 141.102, 140.052, 139.697, 140.003, 138.426,
1158        . 136.868, 136.514, 136.521, 134.938, 133.350, 131.947, 131.400, 130.087,
1159        . 129.236, 129.069, 127.748, 127.234, 126.489, 125.071, 123.494, 121.924,
1160        . 120.215, 118.591, 117.526, 115.774, 114.039, 113.586, 114.065, 114.092,
1161        . 114.092, 113.505, 113.301, 111.648, 110.071, 110.102, 108.491, 106.856,
1162        . 105.614, 106.114, 107.416, 108.808, 109.961, 111.369, 112.897, 113.704,
1163        . 112.361, 112.743, 111.279, 109.102, 107.143, 107.265, 105.033, 105.859/
1164        data (Longitude(i),i= 8401, 8480)/
1165        . 103.540, 101.868, 100.865, 102.036,  99.892,  99.894,  99.811, 100.177,
1166        . 100.187,  99.103,  99.659,  97.567,  95.677,  93.871,  92.429,  90.500,
1167        .  88.789,  86.995,  86.903,  86.734,  87.295,  85.844,  86.783,  86.786,
1168        .  85.802,  87.012,  85.249,  83.477,  81.679,  80.278,  80.812,  81.132,
1169        .  82.382,  83.534,  83.184,  83.598,  83.045,  83.182,  83.180,  83.048,
1170        .  82.195,  83.024,  81.669,  80.686,  79.296,  77.739,  76.281,  76.643,
1171        .  78.056,  76.916,  76.916,  75.529,  75.213,  75.746,  75.483,  75.097,
1172        .  73.980,  73.142,  73.768,  74.334,  73.763,  73.704,  73.889,  75.278,
1173        .  76.681,  77.699,  78.098,  77.473,  78.587,  79.048,  79.048,  77.737,
1174        .  77.226,  77.317,  76.615,  75.246,  74.427,  74.807,  74.310,  73.864/
1175        data (Longitude(i),i= 8481, 8560)/
1176        .  72.961,  72.213,  70.974,  69.743,  69.899,  69.899,  70.804,  71.359,
1177        .  72.320,  73.074,  73.126,  73.111,  72.478,  72.555,  72.609,  72.768,
1178        .  72.446,  72.229,  72.719,  72.798,  71.559,  71.557,  69.851,  68.906,
1179        .  68.566,  67.953,  66.826,  67.321,  67.099,  66.789,  68.096,  68.417,
1180        .  68.812,  68.650,  68.650,  67.541,  66.464,  65.278,  64.190,  62.837,
1181        .  61.395,  60.285,  60.935,  59.881,  58.701,  57.553,  56.488,  56.493,
1182        .  55.154,  54.010,  53.844,  52.472,  51.321,  50.153,  49.064,  47.908,
1183        .  47.752,  46.711,  46.038,  46.041,  45.305,  45.397,  46.707,  46.160,
1184        .  45.064,  43.691,  44.210,  44.122,  43.839,  44.431,  44.097,  42.908,
1185        .  41.741,  41.644,  41.644,  40.603,  39.745,  40.124,  40.519,  39.351/
1186        data (Longitude(i),i= 8561, 8640)/
1187        .  38.188,  37.473,  36.506,  37.145,  38.064,  37.655,  37.655,  36.508,
1188        .  35.748,  34.879,  34.630,  34.698,  34.334,  33.358,  32.562,  31.854,
1189        .  31.854,  32.981,  34.158,  35.243,  36.427,  37.594,  38.832,  39.999,
1190        .  40.850,  41.376,  41.031,  40.129,  39.609,  39.611,  38.538,  37.580,
1191        .  36.452,  35.198,  33.770,  32.403,  32.096,  30.786,  27.861,  28.769,
1192        .  29.734,  29.001,  28.018,  27.256,  26.299,  25.407,  25.404,  24.411,
1193        .  23.563,  23.808,  24.087,  24.297,  24.297,  24.302,  24.398,  23.622,
1194        .  23.078,  22.361,  21.579,  21.290,  21.067,  21.064,  21.241,  21.244,
1195        .  21.102,  20.636,  19.944,  19.625, 122.067, 121.438, 121.153, 117.787,
1196        . 118.392, 118.994, 118.953, 119.429, 119.767, 122.066, 122.089, 121.618
1197        data (Longitude(i),i= 8641, 8720)/
1198        . 121.488, 121.210, 121.854, 122.276, 122.826, 123.412, 124.065, 124.371,
1199        . 117.787, 117.568, 117.892, 118.534, 119.147, 119.431, 119.983, 120.538,
1200        . 120.978, 121.153, 119.600, 119.197, 119.533, 119.920, 120.167, 120.731,
```

119

```
1201      . 121.264, 121.841, 122.469, 122.693, 122.089, 121.461, 120.941, 120.312,
1202      . 119.849, 119.767,  29.661,  29.636,  30.137,  29.715,  30.138,  30.138,
1203      .  30.599,  31.034,  31.189,  31.189,  31.906,  31.738,  31.961,  32.090,
1204      .  32.798,  33.511,  33.010,  33.170,  33.528,  34.007,  34.515,  35.185,
1205      .  35.885,  36.463,  35.765,  35.113,  34.829,  35.175,  35.283,  35.283,
1206      .  35.799,  36.504,  37.193,  37.887,  38.596,  39.308,  38.676,  38.035,
1207      .  37.738,  37.739,  38.224,  38.052,  37.583,  36.869,  37.200,  37.715/
1208      data (Longitude(i),i= 8721, 8800)/
1209      .  38.309,  38.945,  39.431,  39.904,  40.218,  40.218,  40.823,  41.356,
1210      .  41.613,  41.773,  41.403,  40.800,  10.212,  39.998,  67.870,  67.870,
1211      .  88.119,  88.119,  88.144,  88.146,  88.664,  89.664,  88.872,  88.873,
1212      .  73.099,  73.099,  73.124,  73.124,  73.163,  73.163,  73.204,  73.204,
1213      .  73.240,  73.240,  73.247,  73.247,  73.435,  73.435,  72.941,  72.941,
1214      .  72.959,  72.959,  72.984,  72.984,  73.037,  73.037,  73.096,  73.096,
1215      .  73.209,  73.209,  73.379,  73.379,  73.514,  73.514,  73.492,  73.492,
1216      .  73.354,  73.354,  73.170,  73.170,  73.136,  73.136,  73.387,  73.387,
1217      .  73.317,  73.317,  73.252,  73.252,  73.290,  73.290,  73.365,  73.365,
1218      .  73.401,  73.401,  73.518,  73.518,  73.553,  73.553,  73.550,  73.550/
1219      data (Longitude(i),i= 8801, 8880)/
1220      .  73.544,  73.544,  73.556,  73.556,  73.568,  73.568,  72.926,  72.926,
1221      .  72.952,  72.952,  73.104,  73.104,  73.116,  73.116,  73.344,  73.344,
1222      .  73.354,  73.354,  73.315,  73.315,  72.873,  72.873,  72.900,  72.900,
1223      .  73.023,  73.023,  73.009,  73.009,  73.354,  73.354,  73.361,  73.361,
1224      .  73.549,  73.549,  73.559,  73.559,  73.564,  73.564,  73.586,  73.586,
1225      .  72.978,  72.978,  73.584,  73.584,  73.748,  73.748,  72.929,  72.929,
1226      .  72.832,  72.832,  72.794,  72.794,  72.693,  72.693,  72.703,  72.703,
1227      .  72.963,  72.963,  73.345,  73.345,  73.438,  73.438,  73.478,  73.478,
1228      .  73.494,  73.494,  73.508,  73.508,  73.473,  73.473,  73.497,  73.497,
1229      .  73.519,  73.519,  73.596,  73.596,  73.638,  73.638,  73.703,  73.703/
1230      data (Longitude(i),i= 8881, 8960)/
1231      .  73.703,  73.703,  73.377,  73.377,  73.395,  73.395,  72.979,  72.979,
1232      .  73.448,  73.448,  73.049,  73.049,  73.104,  73.104,  73.113,  73.113,
1233      .  73.354,  73.354,  73.323,  73.323,  73.398,  73.398,  73.471,  73.471,
1234      .  73.594,  73.594,  73.630,  73.630,  72.916,  72.916,  72.932,  72.932,
1235      .  72.964,  72.964,  72.977,  72.977,  72.987,  72.987,  73.393,  73.393,
1236      .  73.450,  73.450,  73.429,  73.429,  73.390,  73.390,  73.271,  73.271,
1237      .  73.271,  73.271,  73.242,  73.242,  73.129,  73.129,  73.127,  73.127,
1238      .  73.197,  73.197,  73.188,  73.188,  73.049,  73.049,  72.691,  72.691,
1239      .  73.045,  73.045,  73.070,  73.070,  73.106,  73.106,  73.163,  73.163,
1240      .  73.182,  73.182,  73.214,  73.214,  73.199,  73.199,  72.905,  72.905/
1241      data (Longitude(i),i= 8961, 9040)/
1242      .  72.980,  72.980,  72.913,  72.913,  73.414,  73.414,  72.716,  72.716,
1243      .  72.177,  72.177,  72.113,  72.113,  73.013,  73.013,  72.791,  72.791,
1244      .  72.750,  72.750,  72.549,  72.549,  72.650,  72.650,  73.672,  73.672,
1245      .  73.662,  73.662,  73.647,  73.647,  73.082,  73.082,  72.280,  72.280,
1246      .  72.329,  72.329,  72.299,  72.299,  72.199,  72.199,  72.433,  72.433,
1247      .  71.351,  71.351,  71.261,  71.261,  71.318,  71.318,  71.339,  71.339,
1248      .  71.536,  71.536,  71.586,  71.586,  72.222,  72.222,  72.211,  72.211,
1249      .  71.966,  71.966,  71.867,  71.867,  71.761,  71.761,  71.737,  71.737,
1250      .  71.773,  71.773,  71.835,  71.835,  79.794,  79.836,  79.940,  80.096,
1251      .  80.564,  81.040,  81.496,  81.810,  81.878,  81.690,  81.417,  81.415/
1252      data (Longitude(i),i= 9041, 9120)/
1253      .  81.414,  81.194,  80.892,  80.567,  80.177,  80.554,  80.059,  80.074,
1254      .  79.931,  79.823,  79.794,  79.794,  81.756,  81.757,  79.756,  79.756,
1255      .  79.678,  79.678,  79.882,  79.882,  89.070,  89.069,  92.813,  92.887,
1256      .  92.779,  92.874,  92.766,  92.753,  90.390,  90.390,  90.388,  90.388,
1257      .  90.434,  90.434,  90.593,  90.593,  90.611,  90.611,  90.680,  90.861,
1258      .  90.572,  90.680,  90.621,  90.621,  90.603,  90.603,  90.588,  90.588,
1259      .  90.472,  90.472,  91.460,  91.460,  91.196,  91.196,  91.099,  91.099,
1260      .  90.981,  90.981,  90.994,  90.994,  90.955,  90.955,  91.873,  91.873,
1261      .  91.934,  91.934,  90.666,  90.666,  92.926,  92.931,  93.415,  93.419,
1262      .  93.694,  93.697,  93.638,  93.641,  93.396,  93.397,  93.721,  93.721/
1263      data (Longitude(i),i= 9121, 9200)/
1264      .  94.657,  94.658,  94.824,  94.826,  99.072,  99.072,  97.871,  97.871,
```

```
1265      .  97.889,   97.889,   97.518,   97.519,   97.665,   97.666,   97.808,   97.811,
1266      .  97.919,   97.921,   97.928,   97.930,   94.278,   94.281,   98.284,   98.285,
1267      .  97.837,   97.836,   98.311,   98.312,   98.236,   98.233,   98.470,   98.472,
1268      .  98.623,   98.618,   97.989,   97.985,   98.067,   98.065,   97.844,   97.843,
1269      .  97.650,   97.649,   97.470,   97.465,   98.053,   98.050,   98.470,   98.466,
1270      .  98.412,   98.413,   98.239,   98.237,   98.254,   98.252,   98.166,   98.161,
1271      .  98.244,   98.244,   98.488,   98.489,   98.498,   98.498,   98.430,   98.430,
1272      .  97.891,   97.891,   97.943,   97.941,   98.230,   98.229,   98.234,   98.234,
1273      .  98.061,   98.058,   98.286,   98.284,   98.285,   98.284,   93.040,   93.041/
1274      data (Longitude(i),i= 9201, 9280)/
1275      .  93.063,   93.021,   92.992,   92.771,   92.742,   92.638,   92.724,   92.811,
1276      .  92.956,   93.063,   92.735,   92.733,   93.138,   93.138,   93.020,   93.020,
1277      .  93.873,   93.873,   92.275,   92.280,   92.686,   92.686,   92.559,   92.561,
1278      .  92.785,   92.787,   93.099,   93.098,   93.608,   93.607,   93.534,   93.534,
1279      .  93.558,   93.563,   93.390,   93.391,   93.661,   93.664,   93.695,   93.696,
1280      .  93.863,   93.868,  100.813,  100.814,   98.414,   98.416,   98.325,   98.320,
1281      . 100.683,  100.684,   98.635,   98.635,   98.537,   98.543,   98.329,   98.333,
1282      .  99.400,   99.401,   99.208,   99.214,  100.535,  100.542,  100.081,  100.076,
1283      . 100.036,  100.038,   99.698,   99.699,   99.682,   99.684,   99.661,   99.663,
1284      . 102.242,  102.243,  102.310,  102.309,  102.561,  102.560,   99.838,   99.836/
1285      data (Longitude(i),i= 9281, 9360)/
1286      .  99.852,   99.853,  100.308,  100.308,  104.188,  104.188,  103.980,  103.992,
1287      . 103.028,  103.031,  104.010,  104.012,  106.656,  106.659,  106.441,  106.440,
1288      . 106.767,  106.774,  108.942,  108.943,  112.913,  112.913,  114.374,  114.374,
1289      . 115.431,  115.431,  106.760,  106.760,  106.911,  106.912,  107.343,  107.344,
1290      . 107.559,  107.559,  107.559,  107.555,  107.716,  107.721,  107.839,  107.839,
1291      . 107.742,  107.744,  107.619,  107.620,  107.468,  107.471,  107.794,  107.796,
1292      . 107.870,  107.869,  111.201,  111.201,  111.762,  111.762,  112.741,  112.741.
1293      . 112.333,  112.333,  112.346,  112.346,  112.271,  112.271,  112.209,  112.209,
1294      . 111.701,  111.701,  111.600,  111.600,  111.487,  111.487,  116.707,  116.707,
1295      . 109.083,  109.083,  109.574,  110.107,  110.620,  110.994,  110.671,  110.478/
1296      data (Longitude(i),i= 9361, 9440)/
1297      . 110.100,  109.707,  109.188,  108.705,  108.642,  108.858,  109.308,  109.580,
1298      . 110.451,  110.456,  110.525,  110.532,  118.091,  118.079,  119.701,  119.701,
1299      . 121.897,  121.901,  122.124,  122.123,  121.969,  121.970,  121.201,  121.734,
1300      . 121.195,  121.204,  124.726,  124.732,  126.442,  126.446,  126.379,  126.381,
1301      . 126.834,  126.369,  126.832,  126.840,  127.931,  127.938,  128.722,  128.728,
1302      . 130.916,  130.923,  121.276,  121.831,  121.825,  121.646,  121.520,  121.400,
1303      . 121.129,  120.872,  120.373,  120.164,  120.128,  120.297,  120.557,  120.887,
1304      . 121.286,  121.498,  121.499,  119.525,  119.526,  118.292,  118.294,  131.654,
1305      . 132.000,  131.606,  131.185,  130.718,  130.294,  129.753,  129.963,  130.559,
1306      . 130.592,  134.606,  134.379,  133.792,  133.248,  132.963,  132.451,  132.482/
1307      data (Longitude(i),i= 9441, 9520)/
1308      . 132.826,  133.432,  133.872,  134.466,  134.604,  141.138,  141.403,  141.618,
1309      . 141.876,  142.064,  141.818,  141.468,  141.056,  140.958,  141.021,  140.783,
1310      . 140.615,  140.761,  140.416,  140.044,  140.106,  139.675,  139.100,  138.489,
1311      . 138.006,  137.397,  136.841,  136.883,  136.318,  136.027,  135.466,  135.108,
1312      . 135.321,  134.830,  134.221,  133.668,  133.134,  132.527,  132.158,  131.601,
1313      . 130.996,  131.360,  131.848,  132.304,  132.676,  133.271,  133.880,  134.475,
1314      . 135.083,  135.654,  135.959,  136.363,  136.722,  136.734,  137.288,  136.878,
1315      . 137.266,  137.794,  138.361,  138.768,  139.224,  139.495,  139.805,  140.017,
1316      . 139.977,  139.988,  140.279,  140.913,  140.906,  141.141,  141.954,  142.409,
1317      . 142.814,  143.347,  143.924,  144.578,  145.176,  145.115,  145.458,  144.881/
1318      data (Longitude(i),i= 9521, 9600)/
1319      . 144.193,  143.650,  143.325,  142.651,  142.078,  141.411,  140.726,  140.802,
1320      . 140.434,  140.066,  139.842,  140.363,  140.468,  141.095,  141.368,  141.666,
1321      . 141.786,  141.644,  141.959,  145.313,  145.313,  129.468,  129.473,  129.330,
1322      . 129.331,  129.773,  129.774,  129.537,  129.538,  129.128,  129.123,  135.004,
1323      . 135.000,  139.414,  139.420,  139.778,  139.783,  139.573,  139.572,  141.176.
1324      . 141.180,  140.994,  140.995,  139.446,  139.453,  138.513,  138.313,  138.458.
1325      . 138.511,  133.292,  133.299,  130.592,  130.391,  130.189,  130.228,  130.602,
1326      . 130.764,  131.291,  131.457,  131.621,  131.656,  129.973,  129.971,  128.655,
1327      . 128.652,  131.071,  130.879,  131.049,  131.074,  130.520,  130.524,  129.949,
1329      . 129.949,  129.872,  129.876,  129.714,  129.715,  130.002,  130.006,  129.690/
```

121

```
1329        data (Longitude(i),i= 9601, 9680)/
1330    . 129.217, 129.685, 129.691, 129.190, 129.197, 128.927, 128.934, 128.556,
1331    . 128.563, 128.279, 127.944, 128.255, 128.287, 126.737, 126.747, 131.234,
1332    . 131.238, 125.288, 125.292, 124.091, 124.095, 123.786, 123.792, 122.948,
1333    . 122.955, 131.259, 131.259, 140.321, 140.321, 142.080, 142.080, 142.189,
1334    . 142.189, 142.202, 142.202, 142.236, 142.236, 142.129, 142.129, 141.358,
1335    . 141.358, 141.295, 141.295, 145.273, 145.273, 140.882, 140.882, 146.164,
1336    . 145.882, 146.161, 146.170, 146.174, 146.868, 146.875, 141.832, 141.844,
1337    . 141.694, 141.821, 142.201, 142.092, 142.160, 142.149, 142.081, 141.857,
1338    . 142.150, 142.064, 141.984, 141.914, 141.898, 142.444, 143.178, 143.426,
1339    . 143.534, 143.067, 142.731, 142.548, 142.736, 142.990, 143.462, 144.193/
1340        data (Longitude(i),i= 9681, 9760)/
1341    . 144.650, 144.306, 144.123, 143.893, 143.694, 143.552, 143.371, 143.105,
1342    . 143.326, 143.324, 133.981, 133.982,  60.493,  60.472, 143.324, 143.230,
1343    . 142.901, 142.863, 142.659, 142.014, 141.832, 136.671, 136.668, 137.216,
1344    . 138.049, 137.225, 137.222, 150.455, 150.454, 155.481, 155.479, 148.800,
1345    . 148.236, 147.613, 147.209, 147.495, 147.923, 148.635, 148.800, 149.469,
1346    . 150.083, 149.474, 152.198, 152.207, 152.421, 152.423, 152.977, 152.981,
1347    . 153.242, 153.237, 153.981, 153.974, 154.464, 154.467, 154.594, 154.601,
1348    . 154.355, 154.355, 155.446, 155.448, 155.236, 155.926, 155.213, 155.227,
1349    . 156.402, 156.397, 166.241, 166.358, 166.249, 167.433, 168.056, 167.444,
1350    . 167.440, 163.386, 164.228, 163.391, 163.389, 178.791, 180.004, 181.563/
1351        data (Longitude(i),i= 9761, 9840)/
1352    . 181.683, 180.109, 179.982, 178.859, 178.789, 169.412, 168.084, 168.907,
1353    . 169.409, 162.271, 162.273, 161.693, 161.692, 160.716, 160.719, 137.964,
1354    . 137.963, 135.418, 135.414, 140.588, 140.448, 141.152, 140.198, 141.843,
1355    . 143.527, 142.470, 141.165, 146.507, 146.782, 148.301, 150.213, 148.949,
1356    . 146.995, 146.505, 148.406, 148.397, 139.837, 141.691, 143.621, 142.318,
1357    . 143.950, 144.951, 143.681, 141.801, 140.983, 139.580, 137.497, 137.284,
1358    . 137.782, 139.232, 140.062, 135.447, 135.448, 126.680, 128.199, 126.784,
1359    . 126.676, 128.113, 128.101, 127.318, 129.012, 127.327, 127.323, 128.292,
1360    . 128.292, 126.510, 127.756, 129.094, 127.897, 126.735, 126.341, 126.497,
1361    . 122.438, 124.001, 125.323, 126.286, 126.441, 124.981, 123.257, 123.362/
1362        data (Longitude(i),i= 9841, 9920)/
1363    . 122.437, 124.511, 124.508, 119.703, 119.698, 115.920, 115.917, 112.813,
1364    . 112.815, 113.134, 113.119, 112.022, 112.014, 107.219, 107.219, 106.524,
1365    . 106.520, 106.422, 106.422, 105.993, 105.993,  99.475, 101.654, 104.072,
1366    . 105.239, 103.629, 101.001, 100.288,  99.491,  99.452,  99.459,  99.965,
1367    .  97.894,  94.971,  93.763,  94.650,  96.956,  99.497,  99.456,  95.595,
1368    .  93.062,  92.227,  93.328,  96.091,  97.458,  96.315,  95.595,  90.062,
1369    .  90.068,  79.208,  79.218,  76.617,  76.618,  91.112,  93.686,  91.088,
1370    .  91.112,  89.167,  89.157,  96.521,  96.538,  97.420,  97.420,  96.644,
1371    .  96.644,  96.461,  96.464,  82.654,  82.654,  81.636,  81.650,  82.146,
1372    .  82.150,  82.134,  82.138,  87.014,  87.027,  85.467,  85.471,  86.206/
1373        data (Longitude(i),i= 9921,10000)/
1374    .  86.206,  85.647,  85.641,  84.686,  84.692,  83.968,  83.978,  82.830,
1375    .  82.829,  82.561,  82.565,  79.161,  79.164,  78.664,  78.676,  76.861,
1376    .  76.868,  76.182,  76.157,  76.088,  76.081,  75.304,  75.308,  74.096,
1377    .  74.106,  69.868,  70.618,  70.101,  69.874,  58.950,  58.757,  59.951,
1378    .  58.936,  59.143,  59.125,  48.231,  48.999,  50.194,  49.466,  48.217,
1379    .  48.245,  53.001,  53.013,  51.397,  52.490,  52.799,  53.151,  54.666,
1380    .  56.336,  55.482,  55.374,  55.223,  55.224,  55.866,  56.628,  57.329,
1381    .  55.848,  54.346,  53.641,  52.569,  51.406,  56.985,  55.356,  54.697,
1382    .  54.265,  55.927,  57.413,  57.292,  58.442,  58.183,  58.185,  60.007,
1383    .  60.929,  62.460,  64.262,  66.097,  67.820,  68.888,  67.046,  65.994,/
1384        data (Longitude(i),i=10001,10080)/
1385    .  65.993,  64.086,  61.978,  60.076,  58.565,  57.110,  55.860,  56.974,
1386    .  58.796,  58.800,  57.635,  57.635,  55.773,  55.803,  52.316,  52.319,
1387    .  47.616,  49.154,  47.783,  47.637,  44.953,  47.876,  45.116,  44.966,
1388    .  59.307,  61.596,  60.056,  59.313,  62.551,  65.036,  63.616,  62.557,
1389    .  60.063,  60.101,  56.102,  56.104,  54.007,  54.005,  54.428,  57.395,
1390    .  54.429,  55.471,  55.500,  55.564,  55.556,  56.847,  56.847,  57.909,
1391    .  57.909,  61.765,  61.777,  62.164,  62.194,  42.541,  42.578,  35.798,
1392    .  35.812,  23.183,  23.192,  22.634,  22.640,  23.238,  23.253,  24.014,
```

122

```
1393       .  24.022,  22.994,  22.368,  23.009,  60.471,  60.491,  31.519,  32.168,
1394       .  31.502,  31.519,  50.116,  50.116,  53.493,  53.493,  53.083,  53.083/
1395          data (Longitude(i),i=10081,10160)/
1396       .  59.051,  59.051,  59.782,  59.782,  49.340,  48.928,  48.885,  48.942,
1397       .  49.310,  49.939,  50.374,  50.896,  51.487,  52.111,  52.715,  53.320,
1398       .  53.943,  53.894,  53.804,  53.841,  53.983,  53.882,  73.750,  73.753,
1399       . 103.734, 104.141, 104.478, 104.012, 103.703, 103.735, 100.421, 100.681,
1400       . 100.458, 100.302, 100.435, 109.486, 109.805, 109.641, 109.482, 109.185,
1401       . 108.894, 108.285, 107.592, 106.893, 106.277, 105.803, 105.035, 104.232,
1402       . 104.809, 105.541, 106.010, 106.662, 107.096, 107.643, 108.229, 108.560,
1403       . 108.944, 109.221, 109.507,  74.190,  74.719,  75.449,  76.177,  76.844,
1404       .  77.571,  78.264,  78.831,  78.424,  77.698,  76.981,  76.267,  75.555,
1405       .  74.859,  74.290,  74.086,  74.073,  73.664,  73.641,  74.021,  74.202/
1406          data (Longitude(i),i=10161,10240)/
1407       .  61.452,  61.844,  61.297,  61.049,  61.386,  60.883,  60.177,  34.641,
1408       .  35.554,  35.665,  36.087,  36.359,  35.336,  35.048,  34.289,  34.117,
1409       .  34.993,  35.334,  34.464,  34.640,  38.122,  38.705,  38.321,  37.638,
1410       .  37.120,  37.899,  37.689,  38.138,  30.925,  31.713,  32.550,  32.943,
1411       .  32.519,  31.535,  30.855,  30.447,  30.208,  30.946,  29.328,  29.326,
1412       .  29.329,  29.327,  60.176,  59.976,  59.515,  33.691,  34.411,  34.919,
1413       .  35.286,  34.763,  34.193,  33.691,  49.340,  49.415,  49.737,  50.389,
1414       .  49.938,  49.384,  49.097,  48.725,  48.283,  47.901,  47.535,  47.497,
1415       .  47.388,  46.989,  46.883,  46.883,  47.288,  47.546,  48.240,  48.827,
1416       .  49.248,  49.950,  50.572,  51.225,  51.877,  52.609,  53.207,  53.119/
1417          data (Longitude(i),i=10241,10320)/
1418       .  53.079,  53.078,  52.739,  52.094,  51.384,  50.974,  51.502,  50.802,
1419       .  50.857,  51.112,  51.263,  51.846,  52.521,  52.577,  52.463,  52.764,
1420       .  52.884,  53.473,  53.994,  54.288,  54.754,  54.224,  53.614,  52.966,
1421       .  52.756,  52.762,  53.417,  53.675,  53.148,  53.744,  53.882,  59.515,
1422       .  58.820,  58.525,  58.257,  58.201,  58.384,  58.968,  59.658,  60.346,
1423       .  61.008,  60.967,  61.355,  61.452, 149.161, 149.281, 149.486, 149.701,
1424       . 150.243, 150.732, 150.756, 151.036, 151.397, 151.894, 151.894, 152.128,
1425       . 152.499, 152.913, 152.999, 153.072, 153.157, 153.057, 153.357, 153.439,
1426       . 153.439, 153.567, 153.526, 153.357, 153.265, 153.084, 153.056, 152.939,
1427       . 152.739, 152.544, 152.547, 152.547, 152.125, 151.667, 151.355, 151.096/
1428          data (Longitude(i),i=10321,10400)/
1429       . 150.925, 150.779, 150.455, 150.189, 150.086, 150.078, 150.054, 149.900,
1430       . 149.753, 149.187, 148.553, 147.932, 147.412, 146.978, 146.350, 146.379,
1431       . 146.379, 145.885, 145.434, 144.799, 144.994, 144.397, 144.021, 143.549,
1432       . 143.536, 143.533, 142.964, 142.411, 141.783, 141.162, 140.547, 140.161,
1433       . 139.767, 139.671, 139.671, 139.774, 139.493, 139.052, 139.509, 139.083,
1434       . 138.468, 138.279, 138.278, 138.455, 138.435, 138.157, 137.883, 137.711,
1435       . 137.109, 137.454, 137.491, 137.644, 137.875, 137.920, 137.444, 137.224,
1436       . 136.689, 136.274, 135.889, 135.961, 135.962, 135.454, 135.334, 135.058,
1437       . 134.796, 134.263, 134.283, 133.844, 133.254, 132.746, 132.482, 132.483,
1438       . 131.940, 131.432, 130.844, 130.256, 129.668, 129.079, 128.536, 127.991/
1439          data (Longitude(i),i=10401,10480)/
1440       . 127.420, 126.829, 126.239, 126.003, 148.008, 148.267, 148.289, 148.366,
1441       . 147.938, 147.971, 147.993, 147.993, 147.587, 147.253, 146.756, 146.067,
1442       . 145.719, 145.359, 145.339, 144.983, 144.752, 144.754, 144.645, 145.252,
1443       . 145.784, 146.414, 147.027, 147.664, 148.006, 126.003, 125.507, 124.969,
1444       . 124.409, 124.022, 123.734, 123.198, 122.606, 122.004, 121.401, 120.800,
1445       . 120.198, 119.622, 119.558, 119.556, 118.953, 118.452, 117.940, 117.327,
1446       . 116.716, 116.149, 115.696, 115.118, 114.958, 114.989, 114.998, 115.585,
1447       . 115.651, 115.744, 115.745, 115.556, 115.310, 115.062, 114.939, 114.972,
1448       . 114.797, 114.578, 114.579, 114.370, 114.103, 114.034, 113.769, 113.426,
1449       . 113.265, 113.546, 113.531, 113.954, 114.251, 113.942, 113.661, 113.613
1450          data (Longitude(i),i=10481,10560),
1451       . 113.613, 113.391, 113.454, 113.749, 113.813, 113.699, 113.897, 114.124,
1452       . 114.499, 114.831, 115.332, 115.757, 116.083, 116.085, 116.551, 117.073,
1453       . 117.606, 118.069, 118.597, 119.032, 119.565, 119.714, 119.714, 120.245,
1454       . 120.739, 121.200, 121.519, 121.778, 122.151, 122.192, 122.172, 122.566,
1455       . 122.859, 122.970, 122.969, 123.257, 123.503, 123.810, 123.504, 123.601,
1456       . 124.113, 124.618, 124.610, 124.676, 125.181, 125.182, 125.184, 124.826,
```

123

```
1457        . 125.346, 125.298, 125.601, 125.899, 126.142, 126.653, 126.952, 126.955,
1458        . 127.421, 127.772, 129.135, 128.086, 128.420, 128.939, 129.455, 129.953,
1459        . 129.511, 129.590, 129.824, 130.269, 130.143, 130.144, 130.609, 130.969,
1460        . 131.480, 131.993, 132.482, 132.669, 132.188, 132.699, 133.047, 133.048/
1461        data (Longitude(i),i=10561,10640)/
1462        . 133.466, 133.964, 134.454, 134.959, 135.469, 135.888, 135.733, 136.224,
1463        . 136.404, 136.580, 136.581, 136.910, 136.671, 136.483, 135.978, 136.011,
1464        . 135.795, 135.517, 135.368, 135.368, 135.708, 136.149, 136.524, 137.015,
1465        . 137.460, 137.883, 138.304, 138.817, 139.176, 139.608, 139.781, 139.781,
1466        . 140.296, 140.771, 140.957, 141.234, 141.416, 141.449, 141.633, 141.546,
1467        . 141.600, 141.508, 141.631, 141.632, 141.720, 141.679, 141.946, 142.098,
1468        . 142.144, 142.609, 142.788, 142.866, 143.245, 143.247, 143.272, 143.497,
1469        . 143.591, 143.597, 143.773, 144.282, 144.725, 145.154, 145.341, 145.342,
1470        . 145.235, 145.361, 145.406, 145.728, 145.978, 146.102, 146.009, 146.334,
1471        . 146.334, 146.424, 146.873, 147.395, 147.697, 148.220, 148.683, 148.727
1472        data (Longitude(i),i=10641,10720)/
1473        . 149.157, 149.161, 96.924, 96.924, 96.836, 96.836, 117.592, 118.085,
1474        . 118.578, 118.230, 118.692, 119.166, 118.842, 118.430, 117.925, 117.464.
1475        . 117.464, 117.729, 117.350, 116.845, 116.375, 116.094, 115.836, 115.370,
1476        . 115.151, 115.145, 115.029, 115.020, 114.089, 113.844, 113.488, 113.181,
1477        . 112.784, 112.295, 111.800, 111.415, 111.263, 111.019, 110.517, 110.017,
1478        . 109.650, 109.651, 118.656, 118.660, 118.284, 118.395, 117.341, 117.343,
1479        . 117.093, 117.098, 116.869, 116.873, 115.168, 115.166, 111.312, 111.312,
1480        . 115.146, 115.029, 115.019, 114.548, 114.088, 95.238, 95.743, 96.142,
1481        . 96.641, 97.143, 97.635, 97.968, 98.268, 98.592, 99.007, 99.431,
1482        . 99.826, 100.002, 100.406, 100.826, 100.942, 100.942, 101.430, 101.904/
1483        data (Longitude(i),i=10721,10800)/
1484        . 102.190, 102.520, 103.004, 102.659, 103.140, 103.638, 103.491, 103.271,
1485        . 103.270, 103.398, 103.807, 104.310, 104.452, 104.606, 104.722, 105.209
1486        . 105.632, 105.997, 105.881, 105.904, 105.891, 105.903, 105.835, 105.545,
1487        . 105.544, 105.047, 104.613, 104.111, 103.810, 103.372, 102.958, 102.575,
1488        . 102.259, 101.897, 101.526, 101.251, 100.903, 100.790, 100.576, 100.411,
1489        . 100.411, 100.122, 99.808, 99.488, 99.124, 98.953, 98.778, 98.565,
1490        . 98.138, 97.681, 97.597, 97.250, 96.912, 96.449, 96.099, 95.735,
1491        . 95.405, 95.226, 95.238, 95.116, 95.121, 95.379, 95.380, 95.464,
1492        . 96.022, 96.477, 96.457, 97.341, 97.343, 97.150, 97.148, 98.582,
1493        . 98.578, 98.556, 98.555, 98.502, 98.502, 99.201, 98.783, 98.652/
1494        data (Longitude(i),i=10801,10880)/
1495        . 99.068, 99.286, 99.196, 99.858, 99.859, 100.206, 100.203, 100.467,
1496        . 100.467, 102.381, 102.378, 97.918, 97.415, 97.139, 97.629, 97.914,
1497        . 104.784, 104.789, 105.261, 105.261, 107.482, 107.482, 106.826, 106.826,
1498        . 105.889, 106.174, 106.286, 106.762, 106.738, 106.262, 105.941, 105.632,
1499        . 105.134, 105.380, 105.863, 105.894, 107.709, 108.197, 108.077, 107.592,
1500        . 107.699, 107.709, 103.713, 103.721, 104.208, 104.215, 104.482, 104.485,
1501        . 104.611, 104.615, 104.552, 104.556, 104.419, 104.420, 104.352, 104.357,
1502        . 104.286, 104.281, 104.258, 104.258, 104.854, 104.853, 104.666, 104.667,
1503        . 104.152, 104.153, 103.963, 103.966, 103.938, 103.941, 103.836, 103.836,
1504        . 103.524, 103.526, 103.507, 103.507, 103.435, 103.436, 103.296, 103.297/
1505        data (Longitude(i),i=10881,10960)/
1506        . 103.167, 102.693, 103.171, 103.166, 103.051, 102.549, 103.049, 103.052,
1507        . 102.479, 102.478, 102.495, 102.009, 102.495, 102.495, 101.774, 101.774,
1508        . 105.711, 105.706, 106.235, 106.233, 106.238, 106.239, 106.213, 106.213,
1509        . 107.940, 107.939, 108.066, 108.062, 107.746, 107.746, 108.778, 107.782,
1510        . 108.974, 108.973, 107.527, 107.529, 109.756, 109.756, 108.951, 108.956,
1511        . 115.849, 115.845, 116.270, 116.313, 116.167, 116.271, 116.429, 116.429,
1512        . 118.575, 118.575, 117.548, 117.555, 117.231, 117.235, 117.782, 117.786,
1513        . 117.609, 117.611, 117.685, 117.688, 109.650, 109.287, 109.023, 108.846,
1514        . 103.916, 109.185, 109.312, 109.642, 110.023, 109.908, 110.190, 110.279,
1515        . 110.279, 110.609, 111.088, 111.588, 111.836, 112.322, 112.826, 113.326
1516        data (Longitude(i),i=10961,11040)/
1517        . 113.733, 114.268, 114.546, 114.546, 114.603, 115.089, 115.558, 115.996,
1518        . 116.268, 116.172, 116.605, 116.276, 116.602, 117.032, 117.445, 117.442,
1519        . 117.442, 117.529, 117.614, 119.006, 118.505, 118.992, 118.637, 118.224,
1520        . 117.874, 117.995, 117.648, 117.328, 117.030, 117.521, 117.590, 108.819,
```

124

```
1521      . 108.826, 109.904, 110.361, 110.849, 111.355, 111.86., 112.364, 112.872,
1522      . 113.378, 113.846, 114.349, 114.409, 114.152, 113.645, 113.139, 112.811,
1523      . 112.612, 112.612, 112.127, 111.657, 111.153, 110.688, 110.466, 109.961,
1524      . 109.463, 108.955, 108.552, 108.094, 107.606, 107.152, 107.003, 107.003,
1525      . 106.497, 106.013, 105.822, 105.506, 106.007, 106.492, 106.721, 107.224,
1526      . 107.701, 108.193, 108.689, 109.195, 109.691, 109.906, 112.695, 112.695
1527      data (Longitude(i),i=11041,11120)/
1528      . 115.291, 115.293, 114.479, 114.981, 115.488, 115.305, 114.841, 114.447,
1529      . 114.478, 115.548, 115.548, 116.340, 116.617, 116.197, 116.055, 116.345,
1530      . 117.214, 117.686, 118.188, 117.801, 118.308, 118.809, 119.194, 118.701,
1531      . 118.199, 117.696, 117.200, 116.785, 117.133, 117.215, 119.082, 119.088,
1532      . 119.035, 119.541, 120.048, 120.487, 120.832, 120.377, 119.955, 119.496,
1533      . 119.006, 119.037, 119.451, 119.456, 120.272, 120.779, 121.267, 121.773,
1534      . 122.259, 122.694, 122.345, 121.847, 121.342, 120.842, 120.334, 119.827,
1535      . 120.115, 120.276, 123.082, 123.091, 123.404, 123.908, 123.536, 123.410,
1536      . 123.976, 123.982, 124.472, 124.978, 124.554, 124.475, 121.722, 121.726,
1537      . 122.854, 123.294, 122.954, 122.856, 123.435, 123.440, 123.411, 123.416.
1538      data (Longitude(i),i=11121,11200)/
1539      . 125.096, 125.546, 126.031, 126.511, 126.964, 126.511, 126.0L4, 125.497,
1540      . 125.060, 124.656, 124.461, 124.463, 123.985, 123.667, 123.517, 124.026,
1541      . 124.503, 124.851, 125.096, 125.647, 125.647, 125.968, 126.473, 126.050,
1542      . 125.974, 127.417, 127.416, 127.794, 127.798, 128.617, 128.624, 129.629,
1543      . 129.632, 131.646, 131.632, 131.324, 131.227, 131.557, 131.649, 131.036,
1544      . 131.037, 131.753, 131.756, 120.494, 120.529, 120.465, 120.496, 120.074,
1545      . 119.575, 119.376, 119.514, 119.607, 119.471, 119.016, 118.765, 118.952,
1546      . 119.161, 119.299, 119.352, 119.645, 119.842, 119.843, 119.759, 119.867,
1547      . 120.037, 120.489, 120.822, 121.319, 121.791, 122.289, 122.772, 122.847.
1548      . 122.847, 123.339, 123.833, 124.297, 124.610, 124.992, 125.058, 124.725
1549      data (Longitude(i),i=11201,11280)/
1550      . 124.431, 123.942, 123.441, 123.071, 123.069, 122.568, 122.066, 121.565,
1551      . 121.077, 120.586, 120.176, 119.999, 120.119, 120.523, 120.738, 120.738,
1552      . 121.209, 121.578, 122.077, 122.556, 123.031, 123.392, 122.890, 122.559,
1553      . 122.183, 121.744, 121.341, 121.341, 121.770, 122.014, 122.334, 122.196,
1554      . 122.586, 122.890, 122.406, 122.095, 121.597, 121.530, 121.596, 121.596,
1555      . 121.157, 120.931, 121.080, 120.611, 120.200, 120.409, 120.351, 120.366,
1556      . 120.310, 120.398, 120.074, 125.301, 125.299, 125.452, 125.449, 125.449,
1557      . 125.446, 125.658, 125.654, 126.861, 126.860, 126.741, 126.740, 126.878,
1558      . 126.707, 126.731, 126.879, 121.646, 121.647, 121.914, 121.912, 122.049,
1559      . 122.049, 122.132, 122.133, 122.364, 122.365, 123.554, 123.166, 122.882/
1560      data (Longitude(i),i=11281,11360)/
1561      . 123.384, 123.555, 123.144, 123.142, 123.377, 123.378, 123.853, 123.856,
1562      . 124.421, 124.921, 124.530, 124.425, 125.413, 125.915, 125.438, 125.416,
1563      . 125.955, 126.076, 125.888, 125.959, 123.119, 123.122, 123.044, 123.046,
1564      . 121.977, 121.981, 122.709, 122.630, 122.684, 122.712, 123.076, 123.018,
1565      . 123.094, 122.677, 122.813, 122.845, 123.073, 126.127, 126.629, 127.105,
1566      . 127.004, 126.503, 126.096, 126.130, 128.182, 128.685, 129.179, 129.666,
1567      . 130.147, 130.605, 130.872, 130.368, 129.943, 129.446, 128.952, 128.458,
1568      . 127.956, 128.079, 128.182, 128.030, 128.038, 128.434, 128.441, 128.580,
1569      . 128.594, 128.794, 128.794, 129.876, 129.883, 131.419, 131.420, 131.631,
1570      . 131.634, 131.731, 131.733, 132.738, 132.739, 133.176, 132.961, 133.157
1571      data (Longitude(i),i=11361,11440)/
1572      . 133.178, 134.121, 134.488, 134.198, 134.117, 134.123, 134.511, 134.755,
1573      . 134.622, 134.318, 134.489, 134.516, 127.461, 127.964, 127.533, 127.461,
1574      . 127.559, 127.566, 127.527, 127.832, 127.353, 127.531, 127.235, 127.239,
1575      . 127.243, 127.243, 127.393, 127.404, 127.439, 127.439, 127.339, 127.341,
1576      . 128.276, 128.383, 128.298, 128.277, 128.904, 128.441, 127.945, 127.893,
1577      . 128.286, 128.408, 127.987, 127.666, 127.663, 127.663, 127.729, 127.598,
1578      . 127.415, 127.570, 127.875, 128.005, 128.010, 128.010, 127.694, 128.181,
1579      . 128.514, 128.701, 128.261, 128.680, 128.901, 129.321, 129.326, 130.921,
1580      . 131.298, 130.804, 130.302, 130.767, 130.825, 130.635, 130.637, 130.942,
1581      . 130.849, 130.891, 130.899, 129.875, 129.878, 130.351, 129.969, 130.347
1582      data (Longitude(i),i=11441,11520)/
1583      . 130.354, 131.631, 131.640, 133.325, 133.329, 138.553, 139.0.., 138.884,
1584      . 138.484, 137.979, 137.854, 138.171, 138.570, 138.821, 138.824, 141.008.
```

```
1585      . 140.615, 140.271, 140.048, 139.568, 139.061, 139.045, 138.746, 139.165,
1586      . 139.187, 139.187, 138.734, 138.433, 138.400, 138.071, 137.633, 137.176,
1587      . 136.691, 136.217, 135.746, 135.244, 134.815, 134.815, 134.443, 133.951,
1588      . 133.699, 133.405, 133.007, 132.730, 132.618, 132.195, 132.683, 133.062,
1589      . 133.559, 133.679, 133.679, 133.947, 133.448, 132.948, 132.451, 132.006,
1590      . 131.811, 131.324, 131.285, 131.512, 131.512, 131.968, 132.414, 132.905,
1591      . 133.342, 133.841, 134.135, 134.101, 134.140, 134.359, 134.462, 134.462,
1592      . 134.863, 135.316, 135.747, 136.019, 136.345, 136.834, 137.201, 137.200,'
1593        data (Longitude(i),i=11521,11600)/
1594      . 137.546, 138.049, 138.522, 138.976, 139.434, 139.897, 140.401, 140.862,
1595      . 141.008, 134.243, 134.247, 134.406, 134.413, 134.576, 134.576, 134.892,
1596      . 134.894, 135.123, 135.127, 135.499, 135.997, 136.497, 136.003, 135.515,
1597      . 135.505, 135.457, 135.943, 136.338, 135.838, 135.499, 135.461, 141.007,
1598      . 141.488, 141.947, 142.411, 142.887, 143.384, 143.828, 144.301, 144.747,
1599      . 145.127, 145.565, 145.809, 146.036, 146.522, 146.963, 147.460, 147.813,
1600      . 147.537, 147.033, 146.964, 146.961, 147.137, 147.464, 147.832, 148.204,
1601      . 148.464, 148.797, 149.298, 149.339, 149.842, 150.049, 150.542, 150.391,
1602      . 149.902, 150.079, 150.079, 149.583, 149.083, 148.578, 148.070, 147.591,
1603      . 147.313, 146.901, 146.552, 146.284, 146.264, 146.265, 145.877, 145.386/
1604        data (Longitude(i),i=11601,11680)/
1605      . 144.899, 144.436, 143.946, 143.541, 143.130, 142.623, 143.097, 143.397,
1606      . 142.972, 142.477, 141.971, 141.464, 141.008, 143.227, 143.654, 143.208,
1607      . 143.227, 143.339, 143.339, 150.875, 150.875, 153.204, 153.674, 153.190,
1608      . 153.204, 154.033, 154.033, 152.555, 152.555, 151.069, 151.069, 150.772,
1609      . 151.233, 150.754, 150.772, 150.501, 150.918, 150.440, 150.501, 152.560,
1610      . 152.560, 151.123, 151.123, 150.238, 150.238, 147.131, 147.131, 147.848,
1611      . 147.848, 152.181, 152.397, 152.071, 152.022, 151.521, 151.216, 150.752,
1612      . 150.274, 149.767, 149.318, 148.831, 148.358, 148.815, 149.317, 149.816,
1613      . 150.016, 150.156, 150.659, 151.045, 151.457, 151.607, 152.043, 152.181,
1614      . 150.821, 151.217, 151.682, 152.110, 152.497, 152.871, 153.060, 152.662/
1615        data (Longitude(i),i=11681,11760)/
1616      . 152.375, 151.952, 151.554, 151.131, 150.821, 150.879, 150.879, 149.954,
1617      . 150.454, 149.954, 149.954, 149.577, 149.577, 151.995, 151.995, 151.948,
1618      . 151.948, 152.649, 152.649, 153.228, 153.228, 153.662, 153.662, 154.141,
1619      . 154.141, 154.806, 154.806, 159.520, 159.520, 154.697, 155.137, 155.424,
1620      . 155.829, 155.713, 155.266, 154.999, 154.714, 154.697, 154.621, 154.621,
1621      . 149.519, 149.519, 149.165, 149.165, 147.827, 147.827, 147.392, 146.943,
1622      . 147.421, 147.392, 144.523, 144.523, 144.243, 144.243, 144.516, 144.516,
1623      . 142.832, 142.832, 145.992, 145.992, 145.079, 145.079, 120.878, 120.914,
1624      . 122.291, 122.174, 122.283, 122.466, 122.251, 122.033, 121.551, 121.430,
1625      . 121.589, 121.665, 121.959, 122.377, 122.894, 123.051, 123.539, 123.696/
1626        data (Longitude(i),i=11761,11840)/
1627      . 124.021, 124.092, 123.701, 123.310, 122.927, 122.577, 122.654, 122.188,
1628      . 121.773, 121.356, 120.904, 120.802, 120.628, 120.116, 120.009, 119.895,
1629      . 119.769, 120.250, 120.304, 120.435, 120.352, 120.477, 120.574, 121.087,
1630      . 121.555, 122.085, 122.297, 121.867, 121.872, 122.022, 122.026, 121.974,
1631      . 121.982, 121.523, 121.525, 121.231, 121.232, 121.445, 121.444, 121.967,
1632      . 121.972, 121.962, 121.974, 121.957, 121.963, 124.234, 124.327, 124.126,
1633      . 124.236, 123.263, 123.701, 124.053, 123.581, 123.239, 123.268, 123.649,
1634      . 123.649, 123.003, 123.334, 122.948, 123.012, 122.642, 122.645, 122.278,
1635      . 122.280, 122.140, 122.029, 122.112, 122.145, 121.881, 121.889, 120.367,
1636      . 120.882, 121.349, 121.491, 121.386, 120.916, 120.754, 120.454, 120.378/
1637        data (Longitude(i),i=11841,11920)/
1638      . 120.135, 120.146, 124.307, 124.823, 125.323, 125.443, 125.553, 125.061,
1639      . 124.898, 124.506, 124.275, 124.307, 124.345, 124.946, 125.050, 125.198,
1640      . 125.038, 124.733, 124.709, 124.381, 124.345, 125.671, 125.697, 125.618,
1641      . 125.671, 126.095, 126.095, 125.448, 125.872, 126.183, 126.114, 126.350,
1642      . 126.528, 126.585, 126.334, 126.205, 126.082, 125.848, 125.467, 125.603,
1643      . 125.646, 125.179, 124.701, 124.251, 124.054, 124.071, 123.989, 123.523,
1644      . 123.124, 122.684, 122.377, 122.152, 122.047, 122.204, 122.655, 123.029,
1645      . 123.491, 123.859, 124.328, 124.779, 125.191, 125.446, 125.448, 122.089,
1646      . 122.091, 120.586, 120.590, 121.075, 121.079, 120.181, 120.190, 118.454,
1647      . 118.454, 123.649, 123.655, 124.744, 124.745, 124.440, 124.323, 123.828/
1648        data (Longitude(i),i=11921,12000)/
```

```
1649        . 124.149, 124.444, 123.758, 123.759, 124.059, 124.026, 123.864, 123.577,
1650        . 123.599, 123.851, 124.009, 124.059, 123.537, 123.368, 123.199, 123.251,
1651        . 122.744, 122.410, 122.804, 122.881, 123.169, 123.537, 122.707, 122.707,
1652        . 122.026, 122.464, 122.945, 122.799, 122.401, 121.954, 121.988, 122.062,
1653        . 121.931, 122.028, 119.904, 120.343, 119.871, 119.910, 119.929, 119.929,
1654        . 119.869, 119.869, 119.862, 119.873, 117.326, 117.326, 117.084, 117.086,
1655        . 117.059, 117.058, 119.508, 119.506, 119.600, 119.219, 118.748, 118.571,
1656        . 118.156, 117.820, 117.388, 117.579, 117.981, 118.345, 118.665, 119.032,
1657        . 119.323, 119.428, 119.509, 149.885, 149.885, 150.509, 150.509, 151.205,
1658        . 151.205, 153.054, 153.057, 153.145, 153.241, 153.093, 153.054, 153.411/
1659        data (Longitude(i),i=12001,12080)/
1660        . 153.411, 153.394, 153.394, 146.632, 146.632, 147.341, 147.341, 145.343,
1661        . 145.343, 145.326, 145.326, 137.440, 136.822, 137.190, 137.789, 137.440,
1662        . 136.188, 136.188, 134.468, 134.468, 133.563, 133.563, 143.883, 143.936,
1663        . 143.974, 143.883, 148.101, 147.854, 148.319, 148.101, 148.339, 148.339,
1664        . 148.169, 148.169, 147.352, 147.352, 147.135, 147.135, 145.035, 145.035,
1665        . 144.900, 144.900, 144.740, 144.740, 148.010, 148.010, 113.149, 112.913,
1666        . 113.215, 113.149, 113.053, 113.053, 113.110, 113.110, 115.298, 115.298,
1667        . 124.518, 124.518, 124.344, 124.344, 124.908, 124.908, 125.086, 125.086,
1668        . 130.033, 130.253, 130.596, 130.091, 130.033, 130.942, 130.508, 130.375,
1669        . 130.852, 131.353, 131.236, 130.942, 132.567, 132.567, 135.969, 135.969/
1670        data (Longitude(i),i=12081,12160)/
1671        . 136.183, 136.183, 136.508, 136.508, 136.129, 136.129, 136.153, 136.153,
1672        . 136.200, 136.200, 136.364, 136.603, 136.846, 136.332, 136.364, 135.704,
1673        . 135.704, 136.516, 136.516, 136.657, 136.657, 136.841, 136.841, 137.048,
1674        . 137.048, 139.141, 139.538, 139.158, 139.141, 139.415, 139.415, 142.135,
1675        . 142.135, 142.271, 142.271, 142.146, 142.146, 142.300, 142.300, 146.221,
1676        . 146.221, 146.799, 146.799, 148.881, 148.881, 148.967, 148.967, 149.053,
1677        . 149.053, 132.540, 132.540, 136.728, 136.728, 167.971, 167.971, 159.064,
1678        . 159.064, 174.012, 174.311, 173.737, 173.220, 173.023, 172.378, 172.105,
1679        . 171.879, 171.419, 171.196, 171.034, 170.376, 169.847, 169.518, 169.512,
1680        . 168.889, 168.259, 167.852, 167.285, 167.140, 166.709, 167.051, 167.778/
1681        data (Longitude(i),i=12161,12240)/
1682        . 168.346, 169.072, 169.772, 170.253, 170.760, 170.914, 170.576, 171.173,
1683        . 171.173, 171.342, 171.928, 171.699, 171.080, 171.708, 172.246, 172.940,
1684        . 172.727, 172.042, 172.731, 173.150, 173.517, 173.912, 174.286, 174.019,
1685        . 167.484, 167.484, 167.694, 167.690, 168.002, 167.714, 167.798, 168.012,
1686        . 166.624, 166.623, 166.288, 166.292, 169.210, 169.209, 178.838, 178.833,
1687        . 179.079, 179.077,-176.224,-176.224,-176.655,-176.655, 166.732, 166.736,
1688        . 173.785, 173.780, 176.938, 176.834, 176.452, 176.116, 175.683, 175.010,
1689        . 174.976, 175.211, 174.991, 174.406, 173.840, 174.137, 174.628, 174.724,
1690        . 174.940, 174.620, 175.230, 175.791, 175.918, 176.120, 176.712, 177.335,
1691        . 177.817, 178.458, 178.333, 178.180, 177.879, 177.231, 176.950, 176.948/
1692        data (Longitude(i),i=12241,12320)/
1693        . 174.940, 174.800, 174.589, 174.339, 174.416, 173.917, 174.193, 173.766,
1694        . 173.397, 173.197, 172.889, 173.318, 173.935, 174.418, 174.603, 174.794,
1695        . 174.814, 174.620, 175.526, 175.523, 175.169, 175.169,-178.922,-178.925,
1696        .-178.610,-178.609,-178.538,-178.546,-177.908,-177.913, 172.145, 172.146,
1697        . 168.561, 168.559, 153.666, 153.666, 155.449, 155.449, 157.932, 157.932,
1698        . 157.969, 157.969, 157.156, 157.156, 158.157, 158.157, 158.241, 158.241,
1699        . 162.913, 162.913, 151.854, 151.854, 151.855, 151.855, 151.577, 151.577,
1700        . 149.313, 149.313, 149.195, 149.195, 149.186, 149.186, 150.373, 150.373,
1701        . 150.118, 150.118, 149.665, 149.665, 146.180, 146.180, 143.915, 143.915,
1702        . 143.060, 143.060, 166.936, 166.936, 169.734, 169.585, 171.743, 171.743/
1703        data (Longitude(i),i=12321,12400)/
1704        . 172.765, 172.765, 172.884, 172.884, 172.964, 172.964, 173.003, 173.003,
1705        . 172.968, 172.968, 172.855, 172.855, 173.118, 173.118, 172.999, 172.999,
1706        . 173.831, 173.831, 173.636, 173.636, 174.395, 174.395, 174.473, 174.473,
1707        . 174.738, 174.738, 175.057, 175.057, 175.557, 175.557, 175.975, 175.975,
1708        . 176.409, 176.409, 176.137, 176.137, 176.312, 176.312, 177.288, 177.288,
1709        . 177.151, 177.151, 178.703, 178.703, 178.374, 178.374, 179.199, 179.188,
1710        .-172.202,-172.202,-171.234,-171.234,-171.080,-171.080,-171.710,-171.710,
1711        . 167.196, 167.196, 167.114, 167.114, 175.971, 175.971, 176.784, 176.784,
1712        . 168.764, 168.686, 172.026, 171.948, 176.079, 176.079,-176.460,-176.460,
```

127

```
1713        .-176.633,-176.633, 168.133, 168.133, 153.966, 153.966, 171.925, 171.925/
1714        data (Longitude(i),i=12401,12480)/
1715        . 170.840, 170.840, 168.942, 168.942, 168.678, 168.678, 168.558, 168.558,
1716          167.741, 167.741, 166.833, 166.833, 165.509, 165.509, 166.625, 166.625,
1717        . 168.220, 168.220, 166.812, 166.812, 165.527, 165.527, 165.272, 165.272,
1718        . 162.324, 162.324, 167.509, 167.509, 170.136, 170.136, 170.233, 170.233,
1719        . 169.944, 169.944, 169.856, 169.856, 168.975, 168.975, 171.763, 171.808,
1720        . 171.577, 171.754, 171.261, 171.389, 171.053, 171.233, 167.491, 167.510,
1721        . 155.566, 155.566, 155.736, 155.736, 156.046, 156.046, 157.721, 157.721,
1722        . 156.486, 156.927, 157.278, 156.797, 156.455, 156.486, 156.570, 156.570,
1723        . 156.549, 156.549, 157.091, 157.091, 157.033, 157.033, 157.097, 157.097,
1724        . 157.363, 157.363, 157.454, 157.454, 158.216, 158.216, 157.994, 157.994/
1725        data (Longitude(i),i=12481,12560)/
1726        . 157.509, 157.833, 157.327, 157.509, 159.605, 159.605, 158.466, 158.922,
1727        . 159.370, 159.782, 159.278, 158.841, 158.491, 158.466, 158.387, 158.387,
1728        . 158.276, 158.276, 159.230, 159.230, 159.104, 159.104, 159.811, 159.811,
1729        . 160.155, 160.155, 160.329, 160.329, 159.705, 160.188, 160.654, 160.184,
1730        . 159.702, 159.705, 160.720, 160.932, 161.249, 160.775, 160.675, 160.720,
1731        . 161.364, 161.364, 162.729, 162.729, 161.970, 161.970, 161.314, 161.784,
1732        . 162.230, 161.722, 161.368, 161.314, 159.803, 159.803, 160.014, 160.465,
1733        . 159.982, 160.014, 165.794, 165.794, 166.495, 166.495, 166.740, 166.740,
1734        . 159.251, 159.251, 159.359, 159.359, 159.502, 159.502, 159.724, 159.724,
1735        . 159.969, 159.969, 163.661, 163.661, 164.221, 164.221, 165.007, 164.578/
1736        data (Longitude(i),i=12561,12640)/
1737        . 164.102, 164.310, 164.679, 165.048, 165.480, 165.942, 166.411, 166.910,
1738        . 166.642, 166.233, 165.787, 165.411, 165.007, 167.516, 167.516, 166.533,
1739        . 166.533, 166.419, 166.419, 167.388, 167.388, 168.124, 168.124, 171.314,
1740        . 171.314, 169.894, 169.894, 169.462, 169.462, 169.268, 169.268, 168.526,
1741        . 168.526, 168.190, 168.190, 168.223, 168.223, 168.168, 168.168, 168.123,
1742        . 168.123, 167.710, 167.710, 167.443, 167.443, 167.416, 167.416, 167.651,
1743        . 167.651, 167.314, 167.314, 166.658, 166.658, 166.565, 166.565, 166.772,
1744        . 166.681, 167.201, 167.058, 166.772, 167.155, 167.155, 167.201, 167.201,
1745        . 167.422, 167.165, 167.622, 167.422, 177.074, 177.074,-179.957,-180.005,
1746        .-180.000,-179.957, 178.700, 178.901, 179.404, 179.881, 180.002, 179.998/
1747        data (Longitude(i),i=12641,12720)/
1748        . 179.627, 179.152, 178.700, 178.305, 178.305, 179.384, 179.384, 179.423,
1749        . 179.423, 179.321, 179.321, 178.854, 178.854, 178.000, 177.480, 177.390,
1750        . 177.814, 178.336, 178.623, 178.231, 178.000, 177.584, 177.584, 177.280,
1751        . 177.280, 177.146, 177.146, 177.645, 177.645, 178.119, 178.119, 178.494,
1752        . 178.494, 178.171, 178.171, 179.761, 179.761,-179.864,-179.864, 179.881,
1753        . 179.881,-179.315,-179.315,-179.143,-179.143,-179.143,-179.143,-178.963,
1754        .-178.963,-178.315,-178.315,-178.761,-178.761,-178.493,-178.493,-178.966,
1755        .-178.966,-178.583,-178.583,-178.386,-178.386,-178.202,-178.202,-178.727,
1756        .-178.727, 179.919, 180.001, 179.998, 179.919, 174.631, 174.631,-178.993,
1757        .-178.993,-128.331,-128.332,-124.782,-124.783, 134.648, 134.648, 138.192/
1758        data (Longitude(i),i=12721,12800)/
1759        . 138.192, 144.886, 144.886, 145.290, 145.290, 145.647, 145.647, 145.809,
1760        . 145.809, 145.650, 145.650, 145.794, 145.794, 145.850, 145.850, 145.853,
1761        . 145.853, 145.798, 145.798, 145.688, 145.688, 145.393, 145.393, 144.906,
1762        . 144.906, 132.218, 132.218, 134.161, 134.161, 134.282, 134.282, 134.450,
1763        . 134.450,-176.178,-176.178,-172.519,-172.519,-171.741,-171.741,-178.154,
1764        .-178.154,-175.145,-175.145,-169.927,-169.927,-171.859,-171.859,-170.756,
1765        .-170.756,-169.518,-169.518,-175.669,-175.669,-174.057,-174.057,-174.057,
1766        .-174.057,-175.083,-175.083,-174.300,-174.300,-174.934,-174.934,-157.962,
1767        .-157.962,-159.838,-159.838,-157.964,-157.964,-172.493,-172.493,-171.226,
1768        .-171.226,-170.561,-170.561,-169.694,-169.694,-174.666,-174.666,-174.071
1769        data (Longitude(i),i=12801,12880),
1770        .-174.071,-174.259,-174.259,-165.834,-165.834,-165.422,-165.422,-161.096,
1771        .-161.096,-161.032,-161.032,-160.993,-160.993,-163.170,-163.170,-159.803,
1772        .-159.803,-158.944,-158.944,-158.282,-158.282,-158.117,-158.117,-157.719,
1773        .-157.719,-157.328,-157.328,-151.472,-151.472,-151.458,-151.458,-144.315,
1774        .-144.315,-130.101,-130.101,-152.874,-152.874,-151.797,-151.797,-143.492,
1775        .-143.492,-138.635,-138.635,-139.078,-139.078,-138.970,-138.970,-140.073,
1776        .-140.073,-139.515,-139.515,-140.034,-140.034,-140.654,-140.654,-150.221,
```

```
1777          .-150.221,-150.216,-150.216,-152.400,-152.400,-155.900,-155.900,-154.946,
1778          .-154.946,-160.011,-160.011,-157.192,-157.192,-159.260,-159.260,-160.380,
1779          .-160.380,-134.986,-134.986,-140.604,-140.604,-144.626,-144.626,-146.340/
1780          data (Longitude(i),i=12881,12960)/
1781          .-146.340,-149.588,-149.588,-149.856,-149.856,-151.011,-151.011,-151.045,
1782          .-151.045,-151.746,-151.746,-148.232,-148.232,-149.579,-149.579,-149.537,
1783          .-149.537,-150.657,-150.657,-151.362,-151.362,-149.488,-149.488,-147.670,
1784          .-147.670,-135.457,-135.522,-135.612,-135.628,-136.199,-136.163,-136.374,
1785          .-136.411,-136.542,-136.554,-136.310,-136.463,-136.969,-137.068,-138.348,
1786          .-138.383,-138.424,-138.436,-138.801,-138.837,-138.532,-138.569,-138.928,
1787          .-138.798,-138.928,-138.848,-140.415,-140.420,-140.685,-140.721,-141.238,
1788          .-141.268,-140.670,-140.782,-140.821,-140.972,-140.818,-140.653,-140.674,
1789          .-140.802,-140.107,-140.168,-140.867,-140.905,-142.481,-142.527,-142.542,
1790          .-142.530,-142.519,-142.514,-142.572,-142.636,-143.064,-143.111,-143.411/
1791          data (Longitude(i),i=12961,13040)/
1792          .-143.473,-143.385,-143.549,-143.572,-143.699,-143.815,-143.935,-144.059,
1793          .-144.120,-144.173,-144.207,-144.254,-144.291,-144.372,-144.435,-144.929,
1794          .-144.958,-145.001,-144.989,-145.166,-145.237,-145.054,-145.160,-145.433,
1795          .-145.552,-144.999,-145.000,-145.161,-145.202,-145.227,-145.191,-145.911,
1796          .-145.841,-145.856,-145.899,-145.922,-145.977,-146.072,-146.059,-146.254,
1797          .-146.269,-146.220,-146.205,-146.230,-146.239,-145.953,-145.904,-145.710,
1798          .-145.611,-145.613,-145.482,-145.395,-145.370,-145.189,-145.144,-145.379,
1799          .-145.432,-145.459,-145.511,-145.575,-145.605,-144.970,-144.957,-146.496,
1800          .-146.513,-146.609,-146.644,-146.334,-146.379,-147.227,-147.256,-147.347,
1801          .-147.380,-147.394,-147.449,-147.558,-147.581,-148.041,-148.171,-148.232/
1802          data (Longitude(i),i=13041,13120)/
1803          .-148.265,-148.629,-148.689,-148.269,-148.251,-151.722,-151.780,-152.237,
1804          .-152.268,-152.281,-152.292,-153.920,-153.942,-154.531,-154.530,-145.995,
1905          .-146.037,-136.645,-136.655,-136.753,-136.758,-138.789,-138.816,-139.212,
1806          .-139.221,-139.135,-139.160,-141.501,-141.522,-141.912,-141.944,-141.178,
1807          .-141.221,-141.259,-141.279,-142.259,-142.243,-142.212,-142.225,-142.659,
1808          .-142.607,-143.104,-143.077,-146.274,-146.314,-146.840,-146.865,-147.794,
1809          .-147.813,-147.921,-147.918,-148.193,-148.222,-148.695,-148.706,-147.771,
1810          .-147.743,-144.961,-144.981,-143.555,-143.553, 158.837, 158.837,  51.265,
1811          .  51.708,  52.248,  52.655,  53.146,  53.602,  54.168,  54.690,  55.201,
1812          .  55.677,  56.208,  56.766,  56.809,  56.287,  55.798,  56.273,  56.288/
1813
1814          end


0001    !!s LatDat
0002    c---------------------------------------------------------------------------
Segment LatDat
0003          block data LatDat
0004    c---------------------------------------------------------------------------
0005    c     array of latitude values
0006
0007          include 'LatCom.inc'
0008
0009          data (latitude(i),i=    1,   80)/
0010          .  52.838,  53.232,  53.729,  53.534,  53.926,  54.369,  54.829,  55.283,
0011          .  55.739,  55.923,  69.647,  69.573,  69.254,  68.974,  68.862,  68.792,
0012          .  69.263,  69.557,  69.655,  69.654,  69.873,  70.084,  69.628,  69.417,
0013          .  68.991,  69.316,  69.544,  69.842,  69.954,  69.954,  70.229,  69.824,
0014          .  69.412,  69.849,  69.427,  69.646,  69.823,  48.011,  48.439,  48.930,
0015          .  49.430,  49.416,  49.701,  49.701,  50.163,  49.743,  50.193,  50.673,
0016          .  50.459,  50.961,  50.671,  50.952,  50.951,  50.907,  51.131,  51.617,
0017          .  51.646,  51.700,  52.199,  52.636,  52.310,  52.671,  52.837,  55.922,
0018          .  55.421,  54.944,  55.337,  55.830,  55.858,  56.361,  56.904,  57.088,
0019          .  57.566,  58.041,  58.517,  58.518,  58.374,  58.824,  59.313,  58.814
0020          data (latitude(i),i=   81,  160)/
0021          .  58.324,  58.572,  58.966,  58.512,  58.403,  58.718,  59.090,  59.380,
0022          .  59.543,  59.543,  59.846,  59.770,  59.794,  60.022,  60.089,  59.997,
0023          .  60.291,  60.472,  60.689,  61.012,  61.278,  61.278,  60.857,  60.362,
```

129

```
0024        .   59.926,   59.842,   59.480,   59.313,   59.716,   60.183,   60.686,   60.988,
0025        .   60.937,   61.437,   61.248,   61.012,   60.916,   60.916,   60.502,   60.025,
0026        .   59.660,   59.375,   58.996,   58.551,   58.099,   57.867,   57.554,   57.549,
0027        .   57.306,   56.879,   56.650,   56.328,   55.893,   55.572,   55.521,   55.374,
0028        .   55.138,   55.101,   55.113,   55.112,   55.344,   55.769,   55.991,   55.793,
0029        .   56.292,   56.604,   56.890,   57.298,   57.483,   57.978,   58.475,   58.904,
0030        .   59.152,   59.151,   58.794,   58.774,   58.757,   58.943,   58.795,   58.641/
0031            data (latitude(i),i=  161,   240)/
0032        .   59.066,   59.565,   60.022,   60.522,   60.038,   59.818,   60.048,   60.329,
0033        .   60.328,   60.807,   60.773,   60.951,   61.448,   61.489,   61.957,   62.418,
0034        .   62.877,   63.258,   63.052,   63.214,   63.214,   63.446,   63.705,   64.204,
0035        .   64.700,   64.703,   64.340,   64.586,   64.503,   64.574,   65.011,   65.414,
0036        .   65.761,   65.761,   66.017,   66.138,   66.518,   66.157,   66.059,   66.264,
0037        .   66.696,   66.481,   66.888,   67.033,   67.415,   67.815,   67.835,   67.836,
0038        .   68.116,   68.603,   68.881,   69.066,   69.504,   69.971,   70.261,   70.599,
0039        .   70.192,   70.192,   70.689,   70.804,   71.094,   71.186,   70.825,   70.898,
0040        .   70.812,   70.811,   70.408,   70.499,   70.316,   70.163,   70.068,   70.064,
0041        .   70.014,   69.699,   69.647,   68.281,   67.889,   67.594,   67.526,   67.088/
0042            data (latitude(i),i=  241,   320)/
0043        .   67.144,   66.816,   66.815,   66.948,   66.744,   66.280,   66.121,   65.658,
0044        .   65.936,   65.935,   65.442,   64.948,   65.282,   65.618,   66.074,   66.387,
0045        .   66.387,   66.370,   66.516,   66.088,   66.263,   65.787,   65.467,   64.993,
0046        .   64.764,   64.762,   64.490,   64.076,   63.641,   63.213,   63.002,   63.002,
0047        .   62.993,   63.314,   63.735,   63.754,   63.755,   63.756,   63.345,   63.025,
0048        .   62.672,   62.249,   62.035,   62.216,   62.411,   62.806,   62.905,   63.138,
0049        .   63.138,   63.412,   63.878,   64.197,   64.645,   64.904,   64.902,   64.901,
0050        .   64.399,   64.594,   64.279,   64.369,   64.856,   65.287,   65.320,   64.904,
0051        .   65.357,   65.432,   65.431,   65.451,   65.893,   66.386,   66.706,   66.593,
0052        .   66.131,   66.241,   66.563,   66.563,   66.745,   66.647,   66.742,   67.177/
0053            data (latitude(i),i=  321,   400)/
0054        .   67.661,   67.940,   55.002,   55.279,   55.642,   56.061,   56.558,   57.062,
0055        .   57.550,   58.007,   58.337,   58.455,   58.454,   58.897,   59.272,   59.742,
0056        .   60.243,   60.741,   60.824,   60.823,   61.320,   61.804,   62.305,   62.543,
0057        .   62.379,   62.264,   62.356,   62.175,   61.835,   61.834,   61.680,   61.191,
0058        .   61.019,   61.079,   60.581,   60.080,   60.001,   60.061,   60.062,   59.945,
0059        .   59.478,   58.986,   58.860,   58.386,   57.987,   57.765,   57.765,   58.003,
0060        .   58.470,   58.043,   58.417,   58.821,   58.359,   58.862,   59.305,   59.792,
0061        .   59.792,   60.196,   59.861,   59.379,   59.055,   58.879,   58.878,   58.418,
0062        .   58.165,   57.768,   57.269,   56.951,   56.951,   56.702,   56.298,   55.865,
0063        .   55.601,   55.107,   55.330,   55.327,   54.824,   55.206,   54.798,   54.740/
0064            data (latitude(i),i=  401,   480)/
0065        .   54.284,   54.111,   53.837,   53.723,   53.297,   53.587,   53.714,   53.714,
0066        .   54.041,   54.181,   53.748,   53.622,   53.249,   52.774,   52.291,   51.837,
0067        .   51.549,   69.825,   69.738,   69.401,   69.257,   68.978,   68.881,   68.859,
0068        .   68.482,   68.175,   67.799,   67.799,   67.692,   67.677,   67.829,   67.880,
0069        .   67.540,   67.215,   66.712,   67.124,   67.581,   68.082,   68.104,   68.105,
0070        .   68.430,   68.331,   68.129,   68.627,   68.804,   68.729,   68.257,   68.021,
0071        .   67.802,   67.724,   67.793,   67.793,   67.734,   67.728,   68.199,   68.474,
0072        .   68.016,   67.526,   67.183,   66.698,   67.162,   67.659,   68.007,   68.003,
0073        .   67.541,   67.359,   67.847,   68.317,   68.759,   69.256,   69.759,   69.827,
0074        .   69.827,   69.809,   69.683,   69.241,   68.820,   68.454,   67.955,   67.550/
0075            data (latitude(i),i=  481,   560)/
0076        .   67.053,   66.712,   66.711,   66.419,   66.843,   66.929,   66.612,   66.287,
0077        .   66.516,   66.029,   65.628,   65.333,   65.611,   65.630,   65.630,   65.858,
0078        .   65.906,   65.679,   65.314,   65.291,   64.812,   64.375,   64.007,   64.077,
0079        .   64.129,   64.129,   63.627,   63.649,   63.780,   64.010,   63.764,   63.532,
0080        .   63.162,   62.790,   62.537,   62.153,   49.143,   48.874,   48.476,   48.274,
0081        .   48.394,   48.240,   47.739,   47.384,   47.000,   46.743,   46.742,   46.977,
0082        .   47.333,   47.733,   48.121,   48.439,   48.703,   48.794,   55.272,   55.220,
0083        .   55.081,   54.577,   54.076,   53.574,   53.072,   52.764,   52.762,   52.339,
0084        .   51.978,   51.561,   51.114,   51.298,   50.962,   50.724,   50.724,   51.154,
0085        .   51.615,   51.207,   51.708,   52.151,   52.648,   53.134,   53.248,   53.248/
0086            data (latitude(i),i=  561,   640)/
0087        .   53.743,   54.215,   54.658,   54.852,   55.001,   51.549,   51.440,   51.269,
```

130

```
0088      .  51.019,  50.611,  50.249,  50.206,  50.138,  50.188,  50.189,  50.280,
0089      .  50.294,  50.306,  50.277,  50.291,  50.215,  49.975,  49.505,  49.286,
0090      .  49.144,  48.794,  48.993,  49.188,  49.261,  49.164,  48.833,  48.381,
0091      .  48.069,  48.158,  48.045,  47.998,  47.997,  47.956,  47.681,  47.758,
0092      .  47.285,  46.786,  46.328,  46.150,  46.108,  46.108,  45.729,  45.679,
0093      .  45.885,  45.574,  45.115,  45.046,  45.044,  44.859,  44.676,  44.500,
0094      .  44.212,  43.828,  43.497,  43.549,  43.549,  43.959,  44.448,  44.759,
0095      .  45.100,  45.111,  45.318,  45.387,  45.854,  45.857,  62.146,  61.910,
0096      .  61.480,  60.990,  60.527,  60.036,  59.533,  59.031,  58.607,  58.740/
0097         data (latitude(i),i=  641,   720)/
0098      .  58.743,  58.741,  58.248,  57.765,  57.283,  57.154,  57.228,  56.975,
0099      .  56.847,  56.516,  56.131,  55.903,  55.695,  55.398,  55.259,  55.272,
0100      .  45.856,  45.523,  45.268,  45.103,  45.151,  44.656,  44.477,  44.347,
0101      .  44.346,  44.356,  43.965,  43.734,  43.433,  43.011,  42.519,  42.056,
0102      .  41.778,  41.828,  41.828,  41.559,  41.478,  41.322,  41.277,  41.164,
0103      .  40.949,  40.613,  40.510,  40.508,  40.039,  39.583,  39.168,  39.298,
0104      .  39.791,  39.289,  38.820,  38.452,  38.452,  38.020,  37.551,  37.132,
0105      .  37.631,  38.131,  38.461,  38.959,  39.370,  39.302,  38.801,  38.301,
0106      .  38.416,  38.334,  38.333,  38.145,  37.723,  37.989,  37.616,  37.129,
0107      .  37.254,  36.857,  36.698,  36.546,  36.547,  36.064,  36.537,  36.102/
0108         data (latitude(i),i=  721,   800)/
0109      .  35.696,  35.696,  35.399,  34.908,  34.607,  34.753,  34.752,  34.313,
0110      .  33.894,  33.822,  33.465,  33.151,  33.151,  32.792,  32.524,  32.536,
0111      .  32.037,  31.581,  31.363,  31.362,  30.861,  30.364,  29.874,  29.401,
0112      .  28.948,  28.515,  28.013,  27.864,  27.864,  28.343,  27.854,  27.383,
0113      .  26.921,  26.419,  25.919,  25.461,  25.185,  25.186,  25.139,  25.578,
0114      .  25.939,  26.399,  26.711,  26.711,  26.999,  27.403,  27.881,  28.303,
0115      .  28.807,  29.193,  29.301,  29.301,  29.706,  30.041,  29.908,  29.704,
0116      .  29.969,  30.262,  30.512,  30.511,  30.385,  30.348,  30.707,  30.350,
0117      .  30.383,  30.181,  30.361,  30.303,  30.302,  30.138,  29.681,  29.290,
0118      .  28.927,  28.927,  29.315,  29.305,  29.144,  29.334,  29.335,  29.749/
0119         data (latitude(i),i=  801,   880)/
0120      .  29.576,  29.632,  29.778,  29.747,  29.745,  29.541,  29.699,  29.204,
0121      .  28.848,  28.603,  28.639,  28.476,  28.475,  28.106,  27.613,  27.288,
0122      .  26.814,  26.335,  26.074,  26.074,  25.968,  32.536,  33.003,  33.003,
0123      .  33.424,  33.740,  34.056,  34.127,  34.423,  34.475,  34.454,  34.454,
0124      .  34.925,  35.403,  35.747,  36.163,  36.603,  36.731,  36.731,  37.060,
0125      .  37.518,  38.007,  38.020,  38.142,  38.022,  38.307,  38.307,  38.706,
0126      .  39.156,  39.659,  40.101,  40.560,  41.029,  41.531,  41.790,  41.791,
0127      .  42.279,  42.776,  43.271,  43.740,  44.240,  44.743,  44.920,  44.920,
0128      .  45.422,  45.925,  46.159,  46.326,  46.694,  46.693,  47.114,  47.604,
0129      .  48.058,  48.399,  48.399,  48.190,  48.132,  47.907,  47.506,  47.885/
0130         data (latitude(i),i=  881,   960)/
0131      .  47.384,  47.868,  48.011,  52.647,  52.647,  53.280,  52.820,  53.207,
0132      .  53.280,  53.701,  53.701,  52.816,  52.816,  53.909,  53.546,  53.837,
0133      .  53.909,  53.617,  53.276,  53.546,  53.617,  53.798,  53.798,  54.067,
0134      .  54.067,  55.014,  55.014,  53.229,  52.726,  52.366,  52.654,  53.033,
0135      .  53.229,  54.124,  53.646,  53.168,  53.464,  53.940,  54.110,  53.640,
0136      .  54.022,  54.124,  69.578,  69.578,  69.100,  69.484,  68.996,  69.100,
0137      .  50.697,  50.816,  50.568,  50.121,  50.113,  49.739,  49.377,  48.967,
0138      .  49.055,  49.241,  49.241,  48.805,  48.547,  48.355,  48.832,  49.242,
0139      .  49.463,  49.893,  50.338,  50.465,  50.590,  50.699,  48.604,  48.604,
0140      .  48.711,  48.711,  48.929,  48.929,  49.093,  49.093,  49.792,  49.792/
0141         data (latitude(i),i=  961,  1040)/
0142      .  50.226,  50.226,  50.290,  50.290,  50.431,  50.431,  50.821,  50.821,
0143      .  49.390,  49.390,  51.712,  51.712,  52.048,  52.048,  52.421,  52.005,
0144      .  52.434,  52.421,  52.289,  52.289,  52.183,  52.183,  52.791,  52.791,
0145      .  49.879,  49.879,  62.526,  62.526,  61.924,  61.924,  61.588,  61.588,
0146      .  62.774,  62.774,  67.964,  67.941,  67.975,  67.964,  68.142,  67.640,
0147      .  67.257,  67.372,  67.876,  68.296,  68.233,  68.142,  63.998,  63.998,
0148      .  63.453,  63.453,  63.257,  63.419,  63.275,  63.257,  62.980,  62.491,
0149      .  62.250,  62.744,  62.939,  62.980,  62.411,  61.907,  62.334,  62.411,
0150      .  56.415,  56.415,  56.525,  56.023,  55.893,  56.321,  56.525,  56.499,
0151      .  56.499,  60.809,  60.809,  62.740,  62.740,  60.561,  60.561,  60.485/
```

131

```
0152          data (latitude(i),i= 1041, 1120)/
0153     .  60.485,   56.911,   56.911,   54.912,   54.912,   74.027,   74.276,   73.991,
0154     .  73.604,   73.222,   72.906,   72.570,   72.169,   71.672,   71.434,   71.082,
0155     .  71.512,   71.854,   72.318,   72.498,   72.493,   72.938,   73.438,   73.937,
0156     .  74.416,   74.528,   74.286,   74.026,   72.603,   72.940,   72.815,   72.313,
0157     .  72.456,   72.929,   72.740,   72.264,   71.773,   71.608,   71.612,   72.114,
0158     .  72.606,   73.101,   73.215,   72.838,   72.345,   71.856,   71.379,   71.000,
0159     .  70.999,   70.593,   70.351,   70.075,   69.847,   69.442,   69.223,   68.832,
0160     .  68.863,   68.941,   68.942,   69.149,   69.499,   69.063,   68.879,   68.629,
0161     .  68.623,   68.621,   68.584,   68.467,   68.851,   69.286,   69.281,   69.588,
0162     .  69.995,   69.995,   70.211,   70.307,   70.255,   70.273,   70.545,   70.676/
0163          data (latitude(i),i= 1121, 1200)/
0164     .  70.595,   70.626,   70.998,   71.240,   71.262,   71.262,   71.460,   71.424,
0165     .  71.626,   72.121,   72.601,   72.952,   73.214,   72.955,   72.602,   64.916,
0166     .  64.635,   64.174,   63.983,   63.511,   63.667,   63.981,   63.718,   63.327,
0167     .  63.524,   63.767,   63.767,   63.586,   64.059,   64.560,   65.059,   65.560,
0168     .  65.327,   65.164,   64.914,   68.291,   68.291,   68.484,   68.484,   67.940,
0169     .  67.940,   68.114,   68.114,   68.232,   68.232,   68.092,   68.092,   67.973,
0170     .  67.973,   67.645,   67.645,   67.462,   67.462,   67.196,   67.196,   68.574,
0171     .  68.574,   68.768,   68.768,   68.369,   68.369,   68.996,   68.996,   69.118,
0172     .  69.118,   69.198,   69.198,   69.504,   69.504,   70.660,   70.660,   73.527,
0173     .  73.042,   73.357,   73.762,   73.527,   73.570,   73.570,   69.882,   69.507/
0174          data (latitude(i),i= 1201, 1280)/
0175     .  69.069,   68.573,   68.535,   68.800,   69.299,   69.794,   69.882,   69.546,
0176     .  69.546,   69.566,   69.566,   68.162,   68.162,   69.781,   69.781,   66.251,
0177     .  66.251,   65.832,   65.832,   66.010,   66.010,   66.000,   66.000,   63.477,
0178     .  63.477,   63.543,   63.543,   51.612,   51.161,   50.664,   50.234,   49.781,
0179     .  50.100,   50.008,   50.007,   49.675,   49.493,   49.499,   49.401,   49.254,
0180     .  49.252,   48.886,   48.396,   48.600,   48.226,   47.730,   48.027,   48.171,
0181     .  48.169,   47.699,   47.263,   46.773,   46.980,   47.447,   47.896,   47.419,
0182     .  47.172,   46.866,   46.866,   47.365,   47.501,   47.593,   47.629,   47.667,
0183     .  47.634,   47.633,   47.565,   48.066,   48.419,   48.919,   49.172,   49.559,
0184     .  49.556,   50.022,   50.496,   50.904,   51.360,   51.610,   51.605,   53.179/
0185          data (latitude(i),i= 1281, 1360)/
0186     .  52.746,   52.862,   53.179,   52.092,   52.092,   49.841,   49.666,   49.394,
0187     .  49.171,   49.372,   49.716,   49.841,   50.784,   50.784,   49.634,   49.634,
0188     .  49.736,   49.736,   48.191,   48.191,   47.641,   47.641,   47.071,   47.071,
0189     .  46.851,   46.366,   45.986,   46.202,   45.706,   45.622,   45.924,   46.372,
0190     .  46.816,   46.851,   47.641,   47.298,   47.641,   47.037,   46.537,   46.429,
0191     .  46.478,   45.979,   46.239,   46.398,   46.704,   47.037,   47.918,   47.918,
0192     .  43.939,   43.939,   55.246,   54.792,   55.219,   55.246,   56.350,   56.028,
0193     .  55.636,   55.182,   54.688,   55.122,   55.507,   56.003,   56.350,   55.331,
0194     .  55.331,   56.129,   56.129,   54.986,   54.986,   55.236,   55.236,   55.413,
0195     .  55.413,   55.954,   55.469,   55.794,   55.954,   56.336,   56.336,   56.480/
0196          data (latitude(i),i= 1361, 1440)/
0197     .  56.480,   56.456,   56.456,   56.786,   56.786,   56.978,   56.476,   56.826,
0198     .  56.978,   56.899,   56.411,   56.819,   56.899,   57.431,   56.974,   56.472,
0199     .  56.802,   57.284,   57.431,   57.317,   57.317,   58.249,   58.000,   57.993,
0200     .  57.749,   57.381,   57.784,   58.246,   58.249,   58.089,   58.089,   58.363,
0201     .  58.048,   57.581,   58.027,   57.550,   57.109,   57.590,   58.088,   58.363,
0202     .  60.574,   60.574,   60.393,   60.393,   60.286,   59.865,   60.349,   60.286,
0203     .  58.593,   58.593,   58.360,   58.009,   58.388,   58.360,   58.002,   57.829,
0204     .  57.469,   57.081,   56.989,   57.486,   57.321,   57.817,   58.002,   57.136,
0205     .  57.136,   56.551,   56.551,   57.969,   57.969,   56.604,   56.604,   56.588,
0206     .  56.588,   55.911,   55.911,   55.278,   55.278,   55.238,   55.238,   55.400/
0207          data (latitude(i),i= 1441, 1520)/
0208     .  55.400,   54.965,   54.965,   54.485,   54.485,   54.992,   54.622,   54.471,
0209     .  54.913,   55.042,   54.992,   58.792,   58.792,   39.540,   39.540,   37.931,
0210     .  37.931,   35.200,   35.666,   35.195,   35.200,   35.172,   35.172,   35.049,
0211     .  35.049,   34.915,   34.915,   34.694,   34.694,   34.514,   34.514,   33.893,
0212     .  33.893,   27.843,   27.372,   27.841,   27.843,   25.478,   25.478,   25.328,
0213     .  25.328,   24.867,   24.867,   24.813,   24.813,   24.757,   24.757,   24.717,
0214     .  24.717,   24.745,   24.745,   24.688,   24.688,   24.681,   24.681,   24.591,
0215     .  24.591,   24.552,   24.552,   26.515,   26.515,   29.743,   29.743,   30.397,
```

132

```
0216    .  30.336,  30.404,  30.397,  30.241,  30.241,  30.216,  30.216,  29.980,
0217    .  29.980,  30.116,  30.116,  29.559,  29.559,  29.351,  29.351,  28.476/
0218       data (latitude(i),i= 1521, 1600)/
0219    .  28.476,  28.382,  28.382,  28.106,  28.106,  27.820,  27.381,  27.833,
0220    .  27.820,  27.181,  26.681,  26.195,  26.677,  27.177,  27.181,  24.592,
0221    .  24.592,  44.801,  44.801,  44.439,  44.439,  41.384,  41.384,  41.465,
0222    .  41.465,  41.048,  40.835,  40.708,  40.591,  40.933,  40.980,  41.052,
0223    .  41.048,  45.884,  45.664,  45.833,  45.992,  45.884,  45.990,  45.990,
0224    .  46.104,  46.104,  46.293,  46.293,  45.734,  45.734,  45.276,  44.823,
0225    .  45.300,  45.276,  47.336,  47.336,  48.162,  47.856,  48.180,  48.162,
.226    .  48.839,  48.839,  46.869,  46.869,  46.949,  46.949,  47.062,  47.062,
0227    .  32.993,  32.993,  33.478,  33.478,  33.279,  33.279,  34.084,  34.084,
0228    .  34.033,  34.033,  34.057,  34.057,  48.392,  48.392,  65.163,  65.563/
0229       data (latitude(i),i= 1601, 1680)/
0230    .  65.987,  66.168,  66.156,  66.441,  66.591,  66.862,  67.028,  66.542,
0231    .  66.313,  66.578,  66.076,  65.622,  65.792,  65.301,  64.968,  65.432,
0232    .  65.369,  65.017,  65.163,  53.045,  52.548,  52.125,  51.987,  51.487,
0233    .  51.034,  50.534,  50.862,  51.359,  51.771,  51.771,  52.218,  52.711,
0234    .  53.195,  53.649,  54.108,  53.876,  53.396,  53.052,  53.031,  60.882,
0235    .  60.970,  61.392,  61.537,  61.918,  62.324,  62.487,  62.618,  62.819,
0236    .  62.658,  62.852,  62.852,  62.942,  62.747,  62.326,  62.043,  62.162,
0237    .  62.418,  62.654,  62.231,  61.733,  61.403,  61.202,  60.861,  60.888,
0238    .  43.000,  42.500,  42.043,  41.685,  41.894,  42.374,  42.877,  42.998,
0239    .  44.502,  44.153,  43.654,  43.344,  43.278,  43.388,  43.331,  43.273/
0240       data (latitude(i),i= 1681, 1760)/
0241    .  43.273,  43.264,  43.669,  43.886,  43.976,  43.968,  44.208,  44.365,
0242    .  44.502,  42.478,  42.663,  42.571,  42.849,  42.846,  42.419,  42.278,
0243    .  43.021,  43.512,  43.995,  43.810,  44.311,  44.808,  45.304,  45.502,
0244    .  45.771,  45.773,  45.772,  45.321,  44.835,  44.741,  44.256,  43.757,
0245    .  43.261,  43.000,  42.998,  43.497,  43.993,  44.479,  44.916,  45.325,
0246    .  45.758,  45.809,  45.809,  45.974,  45.969,  45.985,  46.483,  46.498,
0247    .  46.692,  46.685,  46.686,  46.472,  46.658,  46.922,  47.193,  46.888,
0248    .  46.706,  46.584,  46.584,  46.939,  46.751,  47.224,  47.584,  47.793,
0249    .  48.011,  48.011,  48.447,  48.703,  48.935,  48.772,  48.746,  48.626,
0250    .  48.625,  48.155,  47.927,  47.530,  47.039,  46.553,  46.331,  46.284/
0251       data (latitude(i),i= 1761, 1840)/
0252    .  46.284,  46.179,  46.123,  45.979,  45.924,  45.489,  45.066,  44.752,
0253    .  44.752,  44.525,  44.791,  45.193,  44.690,  44.236,  43.741,  43.253,
0254    .  43.015,  42.238,  42.341,  42.280,  42.122,  42.278,  42.041,  41.857,
0255    .  41.606,  41.479,  41.450,  41.710,  42.122,  42.238,  41.967,  42.262,
0256    .  42.478,  41.660,  41.216,  40.756,  41.037,  41.517,  41.660,  66.127,
0257    .  66.378,  66.765,  67.250,  67.689,  68.047,  68.530,  68.105,  68.098,
0258    .  68.293,  68.496,  68.496,  68.588,  68.845,  69.291,  69.588,  69.879,
0259    .  70.172,  70.359,  70.252,  70.068,  70.324,  70.503,  70.439,  70.439,
0260    .  70.740,  70.952,  71.273,  71.502,  71.541,  72.045,  71.720,  71.539,
0261    .  71.207,  70.729,  70.433,  70.709,  71.001,  71.001,  71.502,  71.932/
0262       data (latitude(i),i= 1841, 1920)/
0263    .  72.254,  72.126,  72.568,  72.619,  72.964,  73.104,  73.432,  73.924,
0264    .  73.952,  73.952,  73.563,  73.568,  73.452,  73.954,  74.308,  74.457,
0265    .  74.962,  75.428,  75.554,  75.555,  75.311,  75.724,  75.956,  76.421,
0266    .  76.824,  76.816,  77.306,  77.414,  77.915,  77.523,  77.525,  78.027,
0267    .  78.519,  78.964,  79.468,  79.849,  80.095,  80.231,  80.624,  80.721,
0268    .  81.000,  81.001,  81.418,  81.881,  81.737,  81.535,  81.092,  80.641,
0269    .  81.129,  81.609,  81.594,  82.025,  81.871,  82.186,  82.253,  82.596,
0270    .  82.714,  82.712,  83.064,  83.126,  82.970,  83.174,  83.591,  83.553,
0271    .  83.114,  83.235,  83.200,  83.201,  82.809,  82.362,  82.764,  82.287,
0272    .  81.805,  82.152,  82.512,  82.008,  81.795,  82.280,  82.262,  82.262/
0273       data (latitude(i),i= 1921, 2000)/
0274    .  82.032,  81.599,  81.944,  81.455,  81.136,  80.753,  80.364,  80.062,
0275    .  80.149,  80.146,  79.685,  79.206,  79.066,  78.735,  78.523,  78.021,
0276    .  77.849,  77.506,  77.716,  77.716,  77.368,  77.193,  77.165,  77.203,
0277    .  76.743,  76.273,  76.007,  76.148,  76.373,  76.164,  75.998,  75.998,
0278    .  75.720,  75.218,  74.849,  74.394,  73.902,  73.420,  73.351,  73.351,
0279    .  72.864,  72.453,  71.955,  71.454,  71.438,  71.919,  71.613,  71.482,
```

133

```
0280      .  71.482,   71.166,   71.020,   70.523,   70.649,   70.786,   70.331,   70.017,
0281      .  70.025,   69.531,   69.051,   68.603,   68.549,   68.191,   68.280,   68.587,
0282      .  69.060,   69.303,   69.185,   69.586,   69.524,   69.535,   69.536,   69.635,
0283      .  69.727,   70.198,   69.946,   69.837,   70.236,   70.541,   70.693,   70.477/
0284         data (latitude(i),i= 2001, 2080)/
0285      .  70.946,   70.788,   70.628,   70.628,   71.088,   71.588,   71.212,   71.679,
0286      .  71.306,   71.800,   72.082,   72.582,   72.753,   72.268,   72.128,   72.128,
0287      .  72.309,   71.848,   72.228,   72.027,   72.520,   73.019,   73.491,   73.729,
0288      .  73.461,   72.994,   72.983,   72.982,   73.089,   72.771,   72.959,   72.459,
0289      .  72.097,   72.029,   71.641,   71.142,   71.000,   71.274,   71.689,   72.162,
0290      .  72.659,   72.999,   72.998,   73.458,   73.816,   73.529,   73.092,   72.611,
0291      .  72.111,   71.618,   71.262,   71.072,   70.944,   70.795,   70.420,   70.384,
0292      .  70.010,   69.993,   69.963,   69.922,   69.941,   69.876,   69.876,   70.350,
0293      .  70.239,   69.742,   69.665,   69.210,   68.842,   68.885,   68.514,   68.215,
0294      .  67.942,   67.999,   68.461,   68.940,   69.048,   68.554,   68.989,   69.490/
0295         data (latitude(i),i= 2081, 2160)/
0296      .  69.501,   69.999,   70.153,   70.153,   70.601,   71.053,   71.526,   71.895,
0297      .  71.618,   71.149,   71.118,   71.119,   70.622,   70.131,   69.747,   69.483,
0298      .  68.979,   68.631,   68.249,   68.005,   73.944,   73.802,   73.411,   73.042,
0299      .  72.732,   72.246,   71.760,   71.424,   71.273,   71.273,   71.696,   72.114,
0300      .  72.394,   72.832,   72.693,   73.128,   73.450,   73.847,   73.944,   76.168,
0301      .  75.734,   75.526,   76.028,   76.526,   76.131,   75.761,   75.389,   74.975,
0302      .  75.096,   75.096,   74.872,   74.605,   74.400,   74.655,   74.971,   75.157,
0303      .  75.093,   74.975,   75.122,   75.551,   76.017,   76.456,   76.213,   76.167,
0304      .  76.658,   76.379,   75.923,   75.745,   75.444,   75.403,   75.761,   75.826,
0305      .  75.622,   75.127,   74.897,   74.896,   74.563,   74.511,   74.509,   74.483/
0306         data (latitude(i),i= 2161, 2240)/
0307      .  74.484,   74.552,   74.884,   74.884,   75.187,   75.679,   76.153,   76.314,
0308      .  76.640,   76.973,   76.606,   76.654,   82.465,   82.955,   82.794,   83.113,
0309      .  82.934,   82.640,   82.218,   81.849,   81.621,   81.261,   81.101,   81.101,
0310      .  81.374,   80.877,   80.458,   80.121,   79.828,   79.380,   79.344,   79.366,
0311      .  79.366,   79.237,   79.189,   78.873,   78.394,   77.967,   77.564,   77.303,
0312      .  77.607,   77.158,   77.159,   77.231,   76.902,   76.462,   76.171,   76.554,
0313      .  76.642,   76.371,   76.634,   76.634,   76.446,   76.935,   77.179,   77.632,
0314      .  77.591,   77.369,   77.817,   77.512,   77.974,   78.457,   78.137,   78.634,
0315      .  78.866,   78.599,   78.599,   78.599,   78.959,   79.448,   79.841,   80.333,
0316      .  80.109,   79.685,   80.163,   80.481,   80.583,   80.582,   81.077,   80.699/
0317         data (latitude(i),i= 2241, 2320)/
0318      .  80.747,   80.577,   81.021,   80.653,   81.114,   81.038,   81.528,   81.505,
0319      .  81.506,   81.371,   81.874,   81.934,   82.394,   82.073,   81.834,   82.232,
0320      .  82.728,   82.881,   82.466,   81.111,   80.904,   80.475,   80.087,   79.596,
0321      .  79.334,   78.952,   78.510,   78.333,   78.389,   78.842,   79.162,   79.247,
0322      .  79.247,   79.416,   79.400,   79.874,   80.170,   80.574,   81.049,   81.110,
0323      .  70.944,   71.044,   70.792,   71.063,   70.566,   70.471,   70.900,   71.063,
0324      .  72.819,   72.464,   72.407,   72.837,   72.819,   72.912,   72.869,   72.921,
0325      .  72.912,   73.415,   73.147,   73.045,   73.415,   74.381,   74.381,   75.401,
0326      .  75.401,   82.073,   82.497,   82.113,   82.073,   81.971,   81.971,   70.316,
0327      .  70.086,   69.754,   69.354,   69.354,   69.800,   70.289,   70.316,   69.919/
0328         data (latitude(i),i= 2321, 2400)/
0329      .  69.919,   73.750,   73.633,   73.515,   73.101,   72.892,   72.748,   73.217,
0330      .  73.686,   73.750,   69.442,   69.442,   69.390,   68.909,   69.392,   69.390,
0331      .  69.742,   69.742,   69.790,   69.790,   74.021,   74.170,   74.026,   73.583,
0332      .  73.130,   72.707,   72.765,   72.264,   71.965,   72.467,   72.946,   73.450,
0333      .  73.944,   74.021,   74.100,   73.818,   74.100,   75.631,   75.261,   74.758,
0334      .  74.798,   75.261,   75.631,   75.435,   75.435,   75.934,   75.934,   76.288,
0335      .  76.288,   76.669,   76.669,   76.722,   76.722,   77.749,   77.317,   77.691,
0336      .  77.749,   76.114,   75.634,   76.118,   76.114,   77.361,   77.550,   77.050,
0337      .  76.562,   76.518,   76.036,   75.989,   76.494,   76.914,   77.327,   77.361,
0338      .  76.885,   76.885,   78.099,   77.605,   77.323,   77.756,   78.073,   78.099
0339         data (latitude(i),i= 2401, 2480)/
0340      .  78.059,   78.059,   78.752,   78.295,   78.759,   78.752,   77.893,   77.893,
0341      .  79.353,   78.897,   78.752,   78.266,   78.268,   78.262,   78.720,   79.020,
0342      .  79.353,   77.704,   77.461,   77.704,   78.817,   78.492,   78.040,   77.805,
0343      .  78.280,   78.772,   78.817,   77.654,   77.654,   76.837,   76.837,   76.116,
```

```
0344        .    76.116,    80.151,    79.679,    80.151,    68.706,    68.706,    76.628,    76.591,
0345        .    76.119,    75.616,    75.116,    74.982,    75.412,    75.521,    75.960,    76.144,
0346        .    76.566,    76.628,    75.042,    75.042,    76.748,    76.252,    76.748,    76.748,
0347        .    77.900,    77.900,    77.883,    77.883,    78.938,    78.938,    79.219,    79.219,
0348        .    80.151,    80.151,    77.454,    77.454,    77.456,    77.456,    71.666,    71.666,
0349        .    71.301,    71.301,    71.384,    71.384,    69.417,    69.417,    69.372,    69.372/
0350        data (latitude(i),i= 2481, 2560)/
0351        .    69.876,    69.876,    73.182,    73.182,    73.119,    73.119,    78.668,    78.668,
0352        .    78.857,    78.857,    70.481,    70.481,    66.129,    65.873,    66.323,    65.915,
0353        .    65.572,    65.505,    65.503,    65.003,    64.518,    64.270,    63.989,    63.501,
0354        .    63.522,    63.522,    63.030,    62.875,    62.401,    61.906,    61.594,    61.594,
0355        .    61.126,    60.645,    60.584,    60.323,    60.184,    60.098,    60.502,    60.131,
0356        .    60.594,    60.737,    60.737,    60.514,    61.013,    61.017,    60.919,    61.187,
0357        .    61.188,    61.417,    61.908,    62.263,    62.735,    63.057,    63.057,    63.201,
0358        .    63.622,    64.093,    64.453,    64.340,    64.340,    64.784,    65.166,    64.667,
0359        .    64.283,    64.777,    65.277,    65.728,    65.711,    65.711,    65.695,    65.668,
0360        .    65.988,    65.869,    66.323,    66.662,    66.832,    66.698,    66.355,    66.836/
0361        data (latitude(i),i= 2561, 2640)/
0362        .    67.312,    67.217,    67.187,    67.189,    67.361,    67.314,    67.715,    67.540,
0363        .    67.767,    67.611,    67.940,    67.993,    68.489,    68.236,    68.193,    65.586,
0364        .    65.586,    65.532,    65.532,    65.706,    65.706,    64.476,    64.476,    64.202,
0365        .    64.202,    63.612,    63.612,    63.251,    63.251,    62.790,    62.790,    62.695,
0366        .    62.695,    62.494,    62.494,    61.769,    61.769,    59.966,    59.966,    59.925,
0367        .    59.925,    59.915,    59.915,    59.829,    59.829,    59.988,    59.988,    60.114,
0368        .    60.114,    60.376,    60.376,    60.422,    60.422,    60.434,    60.434,    60.372,
0369        .    60.372,    60.472,    60.472,    60.772,    60.772,    60.746,    60.746,    60.805,
0370        .    60.805,    60.824,    60.824,    60.885,    60.885,    60.862,    60.862,    60.915,
0371        .    60.915,    60.797,    60.797,    60.800,    60.800,    65.574,    65.574,    25.969/
0372        data (latitude(i),i= 2641, 2720)/
0373        .    25.479,    25.031,    24.544,    24.044,    23.543,    23.039,    22.542,    22.048,
0374        .    21.646,    22.031,    21.547,    21.099,    20.754,    20.754,    20.346,    19.954,
0375        .    19.489,    19.063,    18.716,    18.663,    18.364,    18.155,    18.155,    18.295,
0376        .    18.440,    18.485,    18.682,    18.504,    18.565,    19.029,    19.356,    19.374,
0377        .    19.374,    19.854,    20.350,    20.843,    21.196,    21.322,    21.398,    21.572,
0378        .    21.567,    21.446,    21.623,    21.623,    21.201,    20.722,    20.345,    19.853,
0379        .    19.385,    18.882,    18.398,    18.838,    18.486,    18.486,    18.005,    17.506,
0380        .    17.008,    16.520,    16.195,    15.894,    15.894,    15.832,    15.723,    15.723,
0381        .    15.914,    15.814,    15.789,    15.882,    15.944,    15.944,    15.916,    15.876,
0382        .    15.805,    15.805,    15.466,    15.254,    14.994,    14.994,    14.502,    14.023/
0383        data (latitude(i),i= 2721, 2800)/
0384        .    13.550,    13.549,    13.046,    12.544,    12.063,    12.066,    12.065,    11.565,
0385        .    11.069,    10.925,    10.925,    10.463,    10.065,     9.694,     9.566,     7.227,
0386        .     7.615,     8.067,     8.224,     8.128,     8.128,     8.478,     8.665,     8.951,
0387        .     9.141,     9.141,     9.000,     8.933,     8.933,     8.885,     8.885,     8.493,
0388        .     8.308,     7.999,     7.999,     7.627,     7.277,     7.247,     7.732,     7.896,
0389        .     7.896,     8.021,     8.199,     8.351,     8.351,     8.072,     8.028,     9.566,
0390        .     9.186,     8.926,     8.837,     8.837,     8.808,     9.048,     9.201,     9.281,
0391        .     9.274,     9.375,     9.375,     9.560,     9.412,     9.338,     9.221,     9.221,
0392        .     8.857,     8.676,     8.028,     8.481,     8.493,     8.965,     8.965,     9.318,
0393        .     9.515,     9.929,    10.166,    10.279,    10.279,     9.927,     9.852,    10.171/
0394        data (latitude(i),i= 2801, 2880)/
0395        .    10.651,    11.075,    11.075,    11.411,    11.733,    12.149,    12.448,    12.431,
0396        .    12.431,    12.785,    12.986,    12.987,    13.386,    13.406,    13.406,    13.164,
0397        .    13.241,    13.368,    13.368,    13.505,    13.714,    13.746,    13.746,    13.921,
0398        .    13.915,    14.077,    14.368,    14.548,    16.187,    16.322,    15.980,    15.825,
0399        .    15.687,    15.781,    15.934,    16.086,    16.268,    16.526,    16.670,    16.855,
0400        .    17.030,    17.133,    17.132,    17.367,    17.626,    17.978,    18.019,    18.170,
0401        .    18.332,    18.733,    18.988,    19.222,    19.573,    19.831,    19.832,    20.282,
0402        .    20.634,    21.141,    21.604,    22.062,    22.556,    22.930,    23.301,    23.702,
0403        .    24.076,    24.385,    24.730,    25.158,    25.432,    25.680,    26.179,    26.673,
0404        .    26.737,    26.736,    27.128,    27.466,    27.911,    28.178,    28.538,    28.934
0405        data (latitude(i),i= 2881, 2960)/
0406        .    29.354,    29.814,    30.286,    30.737,    31.233,    31.349,    31.349,    31.513,
0407        .    31.771,    31.286,    30.808,    30.309,    29.857,    29.530,    29.143,    28.786,
```

135

```
0408      .  28.366,   27.881,   27.527,   27.118,   27.102,   27.102,   26.618,   26.263,
0409      .  25.769,   25.353,   24.919,   24.415,   24.289,   23.898,   23.520,   23.056,
0410      .  23.272,   23.660,   23.985,   24.281,   24.569,   24.999,   25.499,   25.985,
0411      .  26.094,   26.094,   26.399,   26.701,   26.859,   27.157,   27.491,   27.821,
0412      .  27.775,   28.158,   28.658,   28.991,   29.380,   29.429,   29.427,   29.734,
0413      .  30.219,   30.679,   31.127,   31.533,   32.005,   32.456,   32.536,   16.187,
0414      .  16.031,   15.780,   15.460,   15.154,   15.092,   14.548,   14.906,   15.092,
0415      .  32.312,   32.312,   21.860,   21.860,   21.747,   21.747,   21.855,   21.855/
0416         data (latitude(i),i= 2961, 3040)/
0417      .  21.958,   21.958,   22.413,   22.413,   21.497,   21.497,   21.026,   21.168,
0418      .  20.916,   21.026,   23.097,   23.097,   22.835,   22.835,   22.731,   22.294,
0419      .  22.731,   22.666,   22.666,   23.668,   23.192,   23.668,   23.716,   23.716,
0420      .  24.076,   24.076,   24.644,   24.152,   24.644,   23.456,   23.456,   23.675,
0421      .  23.675,   24.285,   23.804,   24.285,   25.202,   24.811,   24.324,   24.610,
0422      .  25.042,   25.202,   24.179,   24.179,   25.063,   25.063,   25.559,   25.264,
0423      .  24.768,   25.254,   25.559,   26.898,   26.774,   26.390,   25.905,   26.406,
0424      .  26.839,   26.898,   26.686,   26.704,   26.665,   26.686,   18.090,   18.090,
0425      .  18.376,   18.482,   18.464,   18.372,   17.975,   17.973,   17.966,   18.376,
0426      .  18.117,   18.117,   17.771,   17.771,   18.366,   18.366,   18.445,   18.445/
0427         data (latitude(i),i= 3041, 3120)/
0428      .  18.742,   18.742,   18.372,   18.372,   18.164,   18.164,   18.068,   18.068,
0429      .  17.930,   17.930,   17.499,   17.499,   17.402,   17.402,   17.197,   17.197,
0430      .  17.692,   17.692,   17.167,   17.167,   16.814,   16.814,   16.393,   16.393,
0431      .  16.048,   16.048,   15.993,   15.993,   15.628,   15.628,   14.859,   14.512,
0432      .  14.827,   14.859,   19.718,   19.905,   19.911,   19.778,   19.659,   19.336,
0433      .  19.016,   18.846,   18.367,   18.405,   18.404,   18.392,   18.203,   18.308,
0434      .  17.852,   18.034,   18.219,   18.156,   18.228,   18.149,   18.284,   18.567,
0435      .  18.507,   18.445,   18.623,   18.969,   19.469,   19.626,   19.930,   19.776,
0436      .  19.718,   18.195,   18.195,   17.615,   17.615,   18.091,   18.091,   18.625,
0437      .  18.625,   18.932,   18.744,   18.922,   18.932,   20.062,   20.062,   17.417/
0438         data (latitude(i),i= 3121, 3200)/
0439      .  17.417,   17.414,   17.414,   19.326,   19.326,   19.662,   19.662,   18.263,
0440      .  18.512,   18.457,   18.350,   18.162,   17.974,   17.833,   18.016,   18.220,
0441      .  18.263,   21.919,   22.040,   22.512,   22.778,   22.983,   23.030,   23.184,
0442      .  23.157,   23.026,   23.149,   22.937,   22.679,   22.380,   22.344,   22.123,
0443      .  21.799,   21.682,   21.334,   21.241,   21.113,   20.689,   20.685,   20.380,
0444      .  20.200,   20.200,   20.047,   19.888,   19.936,   19.991,   19.937,   19.906,
0445      .  20.382,   20.686,   20.718,   21.040,   21.524,   21.547,   21.689,   21.852,
0446      .  22.057,   22.280,   22.196,   22.677,   22.686,   22.548,   22.221,   22.171,
0447      .  21.863,   21.822,   21.917,   22.800,   22.800,   22.666,   22.666,   22.528,
0448      .  22.528,   22.404,   22.404,   22.309,   22.309,   22.098,   22.098,   20.602/
0449         data (latitude(i),i= 3201, 3280)/
0450      .  20.602,   20.751,   20.751,   20.810,   20.810,   20.894,   20.894,   21.001,
0451      .  21.001,   21.120,   21.120,   21.601,   21.601,   21.838,   21.641,   21.468,
0452      .  21.838,   21.339,   21.339,   21.511,   21.511,   25.442,   25.442,   24.369,
0453      .  24.369,   24.581,   24.581,   25.092,   25.092,   26.065,   26.065,   29.198,
0454      .  29.198,   28.727,   28.727,   28.684,   28.684,   29.584,   29.217,   29.584,
0455      .  31.685,   31.685,   28.917,   28.917,   28.068,   28.068,   25.281,   24.778,
0456      .  25.279,   25.281,   24.530,   24.530,   19.329,   19.329,   18.868,   18.868,
0457      .  18.351,   18.351,   21.797,   21.797,   21.691,   21.691,   18.789,   18.789,
0458      .  20.561,   20.561,   18.719,   18.719,   18.530,   18.530,   10.294,   10.293,
0459      .  17.909,   17.909,   17.300,   17.301,   16.097,   16.097,   16.298,   16.300/
0460         data (latitude(i),i= 3281, 3360)/
0461      .  16.418,   16.418,   13.248,   13.248,   13.303,   13.303,   14.368,   14.369,
0462      .  12.185,   12.185,   11.584,   11.583,    9.435,    9.433,    9.351,    9.349,
0463      .   9.350,    9.349,    8.417,    8.415,    8.305,    8.304,    8.457,    8.453,
0464      .   7.579,    7.578,    7.640,    7.639,    7.290,    7.289,    8.131,    8.131,
0465      .   8.227,    8.227,    8.288,    8.288,   25.426,   24.937,   24.444,   23.943,
0466      .  24.444,   24.944,   25.437,   25.426,   12.114,   11.855,   11.498,   11.147,
0467      .  11.411,   11.889,   12.113,    9.262,    9.305,    9.104,    8.954,    8.675,
0468      .   8.283,    8.663,    8.957,    9.002,    1.947,    2.448,    2.548,    2.859,
0469      .   3.288,    3.706,    4.108,    4.597,    5.096,    5.578,    6.065,    6.500,
0470      .   9.002,    9.404,    9.803,   10.295,   10.729,   11.075,   10.986,   11.345/
0471         data (latitude(i),i= 3361, 3440)/
```

```
0472        .   11.353,   11.356,   11.268,   11.481,   11.738,   12.012,   12.374,   12.355,
0473        .   11.852,    6.500,    6.869,    7.229,    1.437,    1.091,    0.918,    0.719,
0474        .    0.218,   -0.187,   -0.665,   -0.990,   -1.487,   -1.715,   -1.714,   -2.203,
0475        .   -2.537,   -2.403,   -2.859,   -3.291,   -3.374,   -3.374,   -3.691,   -4.075,
0476        .   -4.501,   -4.960,   -5.436,   -5.889,   -6.264,   -6.505,   -6.777,   -7.197,
0477        .   -7.644,   -8.043,   -8.090,   -8.091,   -8.502,   -8.970,   -9.418,   -9.885,
0478        .  -10.346,  -10.773,  -11.264,  -11.592,  -12.074,  -12.444,  -12.870,  -13.306,
0479        .  -13.806,  -13.857,  -13.857,  -14.342,  -14.736,  -15.051,  -15.435,  -15.665,
0480        .  -15.906,  -16.199,  -16.407,  -16.632,  -16.825,  -17.137,  -17.378,  -17.810,
0481        .  -18.104,  -18.349,    1.947,    1.675,    1.437,  -18.348,  -18.846,  -19.346/
0482        data (latitude(i),i= 3441, 3520)/
0483        .  -19.835,  -20.339,  -20.842,  -21.341,  -21.844,  -22.339,  -22.838,  -23.259,
0484        .  -23.747,  -23.843,  -40.813,  -41.315,  -41.771,  -41.502,  -41.491,  -41.857,
0485        .  -42.293,  -42.708,  -43.212,  -43.702,  -44.178,  -44.496,  -44.344,  -44.344,
0486        .  -44.805,  -45.185,  -45.659,  -46.160,  -46.032,  -46.032,  -46.447,  -46.238,
0487        .  -46.278,  -46.278,  -46.554,  -46.794,  -46.838,  -47.339,  -47.664,  -47.838,
0488        .  -47.981,  -48.026,  -48.528,  -48.739,  -48.740,  -49.187,  -49.550,  -50.056,
0489        .  -50.141,  -50.475,  -50.949,  -51.447,  -51.936,  -51.655,  -51.656,  -51.832,
0490        .  -51.509,  -51.726,  -52.226,  -52.219,  -52.672,  -53.102,  -52.699,  -52.535,
0491        .  -52.633,  -52.633,  -52.737,  -53.076,  -53.528,  -53.043,  -52.853,  -53.339,
0492        .  -53.825,  -53.467,  -52.971,  -52.630,  -52.381,  -52.307,  -52.381,  -23.844/
0493        data (latitude(i),i= 3521, 3600)/
0494        .  -24.342,  -24.842,  -25.334,  -25.776,  -26.276,  -26.770,  -27.242,  -27.740,
0495        .  -28.235,  -28.711,  -29.195,  -29.672,  -30.167,  -30.588,  -31.090,  -31.587,
0496        .  -31.612,  -31.612,  -32.113,  -32.613,  -33.079,  -33.582,  -34.024,  -34.516,
0497        .  -35.005,  -35.444,  -35.913,  -36.397,  -36.852,  -37.174,  -37.678,  -38.159,
0498        .  -38.661,  -39.138,  -39.637,  -40.079,  -40.572,  -40.812,  -52.385,  -51.924,
0499        .  -51.592,  -51.114,  -50.609,  -50.234,  -49.742,  -50.113,  -49.708,  -49.210,
0500        .  -48.792,  -48.449,  -48.123,  -47.862,  -47.862,  -47.577,  -47.089,  -47.047,
0501        .  -46.699,  -46.272,  -45.794,  -45.347,  -45.068,  -45.012,  -44.557,  -44.074,
0502        .  -43.572,  -43.304,  -43.304,  -43.059,  -42.782,  -42.510,  -42.864,  -42.433,
0503        .  -42.227,  -42.184,  -41.689,  -41.304,  -41.304,  -40.804,  -40.909,  -41.152/
0504        data (latitude(i),i= 3601, 3680)/
0505        .  -41.160,  -40.960,  -40.474,  -39.976,  -39.517,  -39.056,  -38.989,  -38.976,
0506        .  -38.917,  -38.830,  -38.706,  -38.548,  -38.336,  -37.957,  -37.544,  -37.115,
0507        .  -36.627,  -36.275,  -35.824,  -35.356,  -35.000,  -34.754,  -34.342,  -33.841,
0508        .  -33.339,  -32.912,  -32.448,  -32.448,  -32.935,  -33.348,  -33.851,  -34.241,
0509        .  -34.458,  -34.625,  -34.863,  -34.789,  -34.894,  -34.819,  -34.560,  -34.170,
0510        .  -33.742,  -33.741,  -33.405,  -33.000,  -32.521,  -32.082,  -31.577,  -31.259,
0511        .  -30.816,  -30.376,  -30.749,  -31.071,  -31.509,  -31.871,  -31.624,  -31.266,
0512        .  -30.858,  -30.423,  -29.951,  -29.500,  -29.090,  -28.744,  -28.362,  -27.875,
0513        .  -27.374,  -26.876,  -26.375,  -25.880,  -25.400,  -25.293,  -25.219,  -16.007,
0514        .  -15.508,  -15.006,  -14.506,  -14.012,  -13.518,  -13.048,  -12.942,  -12.560/
0515        data (latitude(i),i= 3681, 3760)/
0516        .  -12.138,  -11.746,  -11.746,  -11.272,  -10.904,  -10.577,  -10.236,   -9.862,
0517        .   -9.489,   -9.085,   -8.916,   -8.916,   -8.445,   -7.967,   -7.462,   -6.961,
0518        .   -6.472,   -5.987,   -5.507,   -5.122,   -5.052,   -5.052,   -5.082,   -4.933,
0519        .   -4.665,   -4.387,   -4.041,   -3.701,   -3.407,   -3.171,   -2.922,   -2.808,
0520        .   -2.858,   -2.882,   -2.881,   -2.906,   -2.823,   -2.679,   -2.459,   -2.489,
0521        .   -2.626,   -2.908,   -3.284,   -2.793,   -2.375,   -1.897,   -1.616,   -1.311,
0522        .   -1.299,   -1.113,  -25.216,  -24.803,  -24.550,  -24.205,  -23.878,  -23.763,
0523        .  -23.601,  -23.391,  -23.353,  -23.354,  -22.944,  -22.898,  -23.010,  -22.967,
0524        .  -22.947,  -22.531,  -22.223,  -22.031,  -21.532,  -21.276,  -21.276,  -20.836,
0525        .  -20.469,  -20.002,  -19.618,  -19.120,  -18.617,  -18.134,  -17.778,  -17.278/
0526        data (latitude(i),i= 3761, 3840)/
0527        .  -16.779,  -16.287,  -16.007,   -1.113,   -1.032,   -0.804,   -0.647,   -0.719,
0528        .   -1.043,   -1.524,   -1.732,   -2.056,   -2.524,   -2.671,   -2.672,   -2.215,
0529        .   -1.898,   -2.020,   -1.783,   -1.782,   -1.289,   -1.256,   -1.405,   -1.405,
0530        .   -1.619,   -1.588,   -1.348,   -1.075,   -1.017,   -1.017,   -0.541,   -0.138,
0531        .    0.174,    0.532,    0.907,    1.325,    1.768,    1.998,    2.464,    2.941,
0532        .    3.430,    3.922,    4.325,    4.033,    4.033,    4.528,    4.823,    5.116,
0533        .    5.465,    5.560,    5.749,    5.347,    5.347,    5.846,    5.949,    5.982,
0534        .    5.976,    5.844,    5.944,    5.824,    5.485,    5.486,    5.978,    6.003,
0535        .    8.577,    8.481,    8.196,    7.957,    7.624,    7.169,    6.687,    6.821,
```

137

```
0536           .    6.527,    6.239,    5.999,   11.852,   11.649,   11.145,   10.692,   10.247'
0537           data (latitude(i),i= 3841, 3920)/
0538           .    9.836,    9.423,    9.052,    9.326,    9.831,   10.249,   10.733,   11.057,
0539           .   11.239,   11.391,   11.880,   12.111,   12.110,   11.800,   11.487,   11.449,
0540           .   11.193,   10.710,   10.454,   10.544,   10.618,   10.604,   10.329,   10.144,
0541           .   10.097,   10.262,   10.446,   10.638,   10.638,   10.769,   10.692,   10.738,
0542           .   10.698,   10.555,   10.055,    9.996,    9.996,    9.891,    9.783,    9.552,
0543           .    9.062,    8.944,    8.627,    8.813,    8.944,    8.627,    8.493,    8.549,
0544           .    8.578,   13.370,   13.368,   12.591,   12.590,   -2.747,   -2.746,  -55.829,
0545           .  -55.830,  -55.824,  -55.824,  -55.587,  -55.590,  -55.098,  -55.199,  -55.068,
0546           .  -55.107,  -54.966,  -55.449,  -55.230,  -55.259,  -54.991,  -54.970,  -54.945,
0547           .  -54.945,  -54.898,  -54.899,  -54.860,  -54.862,  -54.632,  -54.632,  -54.010/
0548           data (latitude(i),i= 3921, 4000)/
0549           .  -54.014,  -53.602,  -54.103,  -53.600,  -53.607,  -54.889,  -54.862,  -54.829,
0550           .  -54.683,  -54.475,  -54.487,  -54.567,  -54.216,  -54.387,  -54.466,  -54.687,
0551           .  -54.687,  -54.210,  -53.819,  -53.542,  -53.435,  -52.944,  -52.540,  -52.556,
0552           .  -52.645,  -53.942,  -53.944,  -54.312,  -54.312,  -54.079,  -54.082,  -54.076,
0553           .  -54.076,  -53.780,  -54.128,  -53.705,  -53.586,  -53.780,  -53.523,  -53.527,
0554           .  -53.397,  -53.400,  -53.252,  -53.256,  -52.727,  -52.990,  -53.275,  -53.063,
0555           .  -52.778,  -52.729,  -52.734,  -52.729,  -52.709,  -52.709,  -52.397,  -52.391,
0556           .  -52.228,  -52.224,  -51.941,  -51.943,  -52.079,  -52.073,  -51.802,  -51.802,
0557           .  -52.307,  -52.309,  -52.088,  -52.088,  -51.900,  -51.892,  -51.613,  -51.614,
0558           .  -51.473,  -51.472,  -51.207,  -51.205,  -51.055,  -51.051,  -50.847,  -50.848/
0559           data (latitude(i),i= 4001, 4080)/
0560           .  -50.889,  -50.889,  -50.794,  -50.792,  -50.434,  -50.431,  -50.050,  -49.554,
0561           .  -49.051,  -49.483,  -49.987,  -50.049,  -49.882,  -49.883,  -49.279,  -49.274,
0562           .  -49.270,  -49.270,  -52.131,  -52.130,  -49.069,  -49.068,  -48.792,  -48.792,
0563           .  -48.694,  -48.198,  -48.671,  -48.692,  -48.438,  -48.436,  -47.977,  -47.978,
0564           .  -48.693,  -48.688,  -48.706,  -48.222,  -48.697,  -48.703,  -48.040,  -48.040,
0565           .  -47.841,  -47.834,  -47.783,  -47.782,  -47.121,  -47.122,  -47.161,  -47.156,
0566           .  -46.102,  -46.097,  -45.737,  -45.730,  -45.756,  -45.757,  -45.971,  -45.972,
0567           .  -45.754,  -45.752,  -45.696,  -45.693,  -45.560,  -45.559,  -45.352,  -45.349,
0568           .  -45.274,  -45.272,  -45.398,  -45.396,  -45.300,  -45.295,  -45.025,  -45.025,
0569           .  -44.964,  -44.958,  -44.820,  -44.818,  -44.911,  -44.907,  -44.694,  -44.696/
0570           data (latitude(i),i= 4081, 4160)/
0571           .  -44.673,  -44.670,  -44.803,  -44.804,  -44.728,  -44.728,  -44.917,  -44.417,
0572           .  -44.915,  -44.918,  -44.527,  -44.526,  -44.558,  -44.556,  -44.439,  -44.433,
0573           .  -44.292,  -44.289,  -44.326,  -44.324,  -44.199,  -44.198,  -43.940,  -43.939,
0574           .  -43.635,  -43.635,  -42.998,  -42.999,  -42.639,  -42.638,  -42.556,  -42.553,
0575           .  -43.386,  -42.947,  -42.446,  -41.957,  -42.173,  -42.596,  -43.052,  -43.380,
0576           .  -55.232,  -55.235,  -55.166,  -55.170,  -54.728,  -54.728,  -54.802,  -54.888,
0577           .  -54.787,  -54.802,  -52.646,  -53.093,  -53.579,  -53.971,  -54.263,  -54.578,
0578           .  -54.634,  -54.898,  -54.938,  -54.887,  -54.832,  -54.787,  -39.088,  -39.088,
0579           .  -27.811,  -27.811,  -26.419,  -26.419,   -0.575,   -0.575,   -0.012,   -0.012,
0580           .   -0.153,   -0.153,    0.201,    0.201,    0.606,    0.606,    0.645,    0.645/
0581           data (latitude(i),i= 4161, 4240)/
0582           .    0.311,    0.013,    0.309,    0.311,    0.071,    0.071,   -0.271,   -0.754,
0583           .   -1.186,   -1.480,   -1.733,   -1.768,   -1.655,   -1.174,   -0.670,   -0.207,
0584           .   -0.138,   -0.240,   -0.160,   -0.234,   -0.271,    0.873,    0.873,    2.181,
0585           .    2.181,    2.257,    2.257,   -1.020,   -0.576,   -1.068,   -1.409,   -1.020,
0586           .   -3.866,   -3.866,  -23.966,  -23.966,  -23.218,  -23.218,   12.626,   12.626,
0587           .   12.384,   12.062,   12.360,   12.384,   12.307,   12.307,   11.823,   11.823,
0588           .   11.817,   11.817,   10.966,   10.966,   11.898,   11.898,   10.806,   10.806,
0589           .   11.063,   11.041,   10.955,   11.063,   10.824,   10.335,   10.067,   10.039,
0590           .   10.274,   10.775,   10.824,   11.343,   11.343,   12.228,   12.228,   13.144,
0591           .   13.144,   13.718,   13.718,   13.057,   13.057,  -60.735,  -60.735,  -60.771
0592           data (latitude(i),i= 4241, 4320)/
0593           .  -60.771,  -54.512,  -54.512,  -54.890,  -54.509,  -54.270,  -54.164,  -54.572,
0594           .  -54.890,  -56.316,  -56.316,  -56.727,  -56.727,  -57.086,  -57.086,  -57.819,
0595           .  -57.819,  -58.493,  -58.493,  -59.070,  -59.070,  -59.469,  -59.469,  -20.490,
0596           .  -20.490,  -20.494,  -20.494,    0.936,    0.936,    0.922,    0.922,  -52.022,
0597           .  -51.786,  -51.413,  -51.915,  -52.208,  -52.025,  -52.110,  -51.717,  -51.335,
0598           .  -51.379,  -51.817,  -52.028,  -52.101,  -51.900,  -51.898,  -27.187,  -27.187,
0599           .  -26.463,  -26.463,  -33.772,  -33.772,  -33.662,  -33.662,  -26.347,  -26.347,
```

138

```
0600      . -26.263,  -26.263,   -1.307,   -1.307,   -1.344,   -1.344,   -0.691,   -0.691,
0601      .  -0.499,   -0.489,    0.364,    0.364,    0.648,    0.648,    0.099,   -0.358,
0602      .  -0.823,   -0.964,   -0.464,   -0.024,    0.099,   -0.289,   -0.289,   -0.149/
0603      data (latitude(i),i= 4321, 4400)/
0604      .  -0.149,  -15.394,  -15.727,  -16.158,  -16.235,  -15.896,  -15.425,  -15.395,
0605      . -69.596,  -69.262,  -68.917,  -68.430,  -68.084,  -67.888,  -67.625,  -67.220,
0606      . -67.149,  -67.146,  -67.259,  -66.910,  -66.503,  -66.734,  -66.162,  -65.914,
0607      . -65.775,  -66.029,  -66.445,  -66.354,  -66.708,  -66.954,  -66.914,  -66.821,
0608      . -66.630,  -66.576,  -66.268,  -66.337,  -66.503,  -66.644,  -66.746,  -66.925,
0609      . -66.882,  -67.219,  -67.237,  -70.427,  -70.651,  -70.544,  -70.601,  -70.793,
0610      . -71.202,  -71.533,  -71.951,  -72.373,  -72.878,  -73.329,  -73.448,  -76.226,
0611      . -76.723,  -77.218,  -77.681,  -78.174,  -78.483,  -78.578,  -78.949,  -79.014,
0612      . -67.625,  -67.221,  -67.213,  -73.700,  -73.666,  -73.477,  -73.284,  -73.174,
0613      . -72.869,  -72.459,  -71.956,  -71.486,  -70.981,  -70.489,  -70.031,  -69.534/
0614      data (latitude(i),i= 4401, 4480)/
0615      . -69.323,  -66.288,  -65.932,  -65.539,  -65.046,  -65.120,  -65.121,  -64.714,
0616      . -64.496,  -64.037,  -63.867,  -63.637,  -63.356,  -63.491,  -63.675,  -64.044,
0617      . -64.368,  -64.665,  -65.035,  -65.427,  -65.911,  -65.971,  -66.392,  -66.256,
0618      . -66.169,  -66.171,  -66.617,  -67.009,  -67.466,  -67.968,  -68.313,  -74.018,
0619      . -74.063,  -73.583,  -73.186,  -72.788,  -72.637,  -72.256,  -71.755,  -71.407,
0620      . -70.933,  -71.256,  -71.750,  -71.365,  -70.934,  -71.380,  -71.420,  -71.417,
0621      . -71.314,  -71.250,  -71.360,  -71.364,  -  .088,  -70.840,  -70.691,  -70.605,
0622      . -70.400,  -70.473,  -70.357,  -70.357,  -70.698,  -70.460,  -70.221,  -70.290,
0623      . -70.160,  -69.971,  -70.153,  -70.367,  -70.466,  -70.509,  -70.358,  -70.197,
0624      . -70.077,  -70.051,  -70.050,  -69.838,  -69.786,  -69.504,  -69.116,  -68.637/
0625      data (latitude(i),,i= 4481, 4560)/
0626      . -68.710,  -69.116,  -84.890,  -85.241,  -85.416,  -85.468,  -85.107,  -85.249,
0627      . -84.812,  -84.628,  -84.286,  -83.806,  -83.307,  -82.812,  -82.317,  -81.885,
0628      . -81.389,  -81.118,  -81.357,  -81.361,  -75.705,  -75.705,  -75.258,  -75.230,
0629      . -75.192,  -75.035,  -75.065,  -75.206,  -74.712,  -74.348,  -73.925,  -73.736,
0630      . -73.617,  -73.617,  -73.356,  -73.338,  -72.950,  -73.064,  -73.044,  -72.999,
0631      . -73.276,  -73.230,  -73.173,  -73.243,  -73.325,  -73.324,  -73.005,  -73.158,
0632      . -73.041,  -73.539,  -73.654,  -73.892,  -73.485,  -73.101,  -73.434,  -73.555,
0633      . -73.799,  -69.322,  -69.171,  -68.904,  -67.082,  -66.604,  -66.288,  -68.314,
0634      . -68.742,  -68.379,  -68.874,  -69.354,  -69.805,  -70.274,  -70.719,  -70.719,
0635      . -71.072,  -71.533,  -72.008,  -72.234,  -72.736,  -73.194,  -73.127,  -73.565/
0636      data (latitude(i),i= 4561, 4640)/
0637      . -74.010,  -74.129,  -74.129,  -74.571,  -74.907,  -75.408,  -75.740,  -76.047,
0638      . -76.246,  -76.594,  -76.668,  -76.580,  -76.360,  -75.959,  -75.929,  -75.930,
0639      . -76.407,  -76.852,  -77.279,  -77.486,  -77.979,  -78.004,  -77.894,  -77.651,
0640      . -78.131,  -78.401,  -78.819,  -78.693,  -78.289,  -78.795,  -79.175,  -79.625,
0641      . -79.626,  -79.279,  -79.500,   79.980,  -79.904,  -80.149,  -80.510,  -80.894,
0642      . -81.118,  -81.411,  -81.773,  -82.123,  -82.454,  -82.952,  -83.038,  -83.040,
0643      . -82.619,  -82.212,  -81.978,  -81.821,  -81.887,  -81.487,  -81.204,  -80.924,
0644      . -80.757,  -80.565,  -80.565,  -80.317,  -79.820,  -79.323,  -79.335,  -79.300,
0645      . -78.986,  -78.482,  -78.035,  -77.621,  -77.344,  -76.991,  -76.624,  -76.403,
0646      . -76.295,  -76.295,  -76.083,  -75.986,  -75.925,  -75.797,  -75.304,  -74.853/
0647      data (latitude(i),i= 4641, 4720)/
0648      . -74.501,  -74.136,  -74.017,  -68.905,  -68.425,  -67.926,  -67.513,  -67.030,
0649      . -67.084,  -69.201,  -69.681,  -69.928,  -70.419,  -70.331,  -70.833,  -71.247,
0650      . -71.728,  -72.231,  -72.371,  -72.021,  -71.589,  -71.090,  -70.680,  -70.188,
0651      . -69.858,  -69.611,  -69.595,  -66.734,  -66.670,  -66.537,  -66.631,  -66.632,
0652      . -66.591,  -66.630,  -66.652,  -66.703,  -66.479,  -66.099,  -65.936,  -65.979,
0653      . -66.004,  -66.004,  -66.170,  -66.374,  -66.403,  -66.719,  -66.686,  -66.283,
0654      . -66.162,  -66.576,  -66.748,  -66.512,  -66.433,  -66.434,  -66.905,  -67.054,
0655      . -66.701,  -66.293,  -66.206,  -66.133,  -66.080,  -65.591,  -65.089,  -65.370,
0656      . -65.871,  -66.230,  -66.268,  -67.237,  -67.626,  -67.811,  -67.847,  -68.293,
0657      . -68.420,  -68.762,  -68.768,  -68.909,  -68.430,  -68.804,  -69.169,  -69.159,
0658      data (latitude(i),i= 4721, 4800),
0659      . -69.305,  -69.619,  -69.979,  -70.242,  -70.427,  -73.448,  -73.514,  -73.827,
0660      . -74.189,  -74.673,  -74.961,  -75.376,  -75.871,  -76.226,  -79.012,  -79.516,
0661      . -79.963,  -80.452,  -80.669,  -81.170,  -81.641,  -82.082,  -82.550,  -82.999,
0662      . -83.385,  -83.798,  -84.051,  -84.306,  -84.442,  -84.457,  -69.116,  -69.542,
0663      . -69.317,  -69.787,  -69.643,  -69.158,  -68.725,  -68.449,  -68.139,  -67.980,
```

139

```
0664        . -67.721, -67.475, -67.729, -67.496, -66.993, -67.057, -67.140, -67.136,
0665        . -66.715, -66.334, -65.994, -65.896, -65.886, -65.997, -66.396, -66.683,
0666        . -67.061, -67.413, -67.374, -67.531, -67.578, -67.578, -67.682, -67.778,
0667        . -67.859, -67.738, -68.231, -68.734, -69.201, -84.457, -84.669, -84.892,
0668        . -81.361, -80.909, -80.493, -80.009, -79.667, -79.422, -79.153, -78.849/
0669          data (latitude(i),i= 4801, 4880)/
0670        . -78.425, -78.158, -77.926, -77.684, -77.684, -77.185, -77.082, -77.227,
0671        . -77.387, -77.761, -77.454, -77.155, -76.654, -76.489, -76.414, -75.912,
0672        . -75.678, -75.505, -75.505, -75.490, -75.112, -75.025, -74.729, -74.745,
0673        . -74.732, -74.829, -74.721, -74.649, -74.649, -74.667, -74.726, -74.723,
0674        . -74.627, -74.344, -74.525, -74.189, -74.444, -74.856, -74.379, -74.700,
0675        . -75.149, -75.257, -77.131, -77.131, -77.759, -77.569, -77.345, -77.836,
0676        . -77.759, -73.421, -73.421, -67.448, -67.448, -66.623, -66.623, -66.263,
0677        . -66.263, -65.682, -65.682, -72.606, -73.102, -72.618, -72.606, -72.981,
0678        . -73.481, -73.290, -72.818, -72.981, -73.191, -73.191, -67.779, -67.779,
0679        . -67.538, -67.082, -66.661, -67.153, -67.571, -67.538, -66.302, -66.302/
0680          data (latitude(i),i= 4881, 4960)/
0681        . -65.864, -65.864, -64.883, -64.883, -64.695, -64.305, -64.806, -64.695,
0682        . -64.428, -64.428, -62.638, -62.462, -62.651, -62.638, -62.521, -62.521,
0683        . -62.361, -62.361, -62.279, -62.279, -62.196, -61.926, -62.220, -62.196,
0684        . -61.243, -61.243, -61.260, -61.260, -63.162, -63.162, -63.426, -63.123,
0685        . -63.430, -63.426, -63.583, -63.583, -64.442, -63.958, -64.072, -64.442,
0686        . -64.528, -64.528, -65.254, -65.254, -70.574, -70.574, -70.448, -70.448,
0687        . -71.041, -71.041, -70.629, -70.629, -70.162, -70.162, -70.156, -70.156,
0688        . -72.651, -72.594, -72.674, -72.651, -72.170, -71.938, -72.054, -71.895,
0689        . -72.177, -72.520, -72.459, -72.265, -72.170, -73.048, -73.048, -73.418,
0690        . -73.418, -73.104, -72.602, -72.824, -73.179, -73.104, -71.178, -70.857/
0691          data (latitude(i),i= 4961, 5040)/
0692        . -70.624, -71.053, -71.191, -71.178, -70.132, -70.132, -69.738, -69.738,
0693        . -72.684, -72.414, -71.942, -71.941, -72.067, -71.578, -71.643, -71.162,
0694        . -71.171, -70.695, -70.237, -69.738, -69.273, -68.882, -69.085, -69.555,
0695        . -70.043, -70.511, -71.007, -71.509, -72.003, -72.457, -72.634, -72.722,
0696        . -72.684, -63.890, -63.890, -68.736, -68.736, -69.666, -69.166, -69.651,
0697        . -69.666, -70.642, -70.642, -71.032, -71.032, -79.557, -79.060, -78.698,
0698        . -78.449, -78.902, -79.348, -79.557, -80.806, -80.429, -80.200, -80.289,
0699        . -80.002, -80.448, -80.806, -80.919, -80.494, -80.140, -79.698, -79.203,
0700        . -78.706, -78.251, -77.936, -78.088, -78.542, -79.043, -79.539, -80.040,
0701        . -80.467, -80.681, -80.852, -80.919, -78.658, -78.602, -78.629, -78.658/
0702          data (latitude(i),i= 5041, 5120)/
0703        . -74.858, -74.485, -74.802, -74.858, -73.137, -73.137, -62.998, -62.998,
0704        . -70.007, -70.007, -72.253, -72.253, -70.556, -70.556, -66.247, -66.247,
0705        . -65.679, -65.679, -65.465, -65.465, -66.872, -66.872, -67.625, -67.625,
0706        . -79.878, -79.717, -79.344, -78.852, -79.054, -79.462, -79.878, -77.126,
0707        . -77.126, -74.464, -74.464, -74.544, -74.544, -74.519, -74.519, -74.164,
0708        . -73.919, -73.643, -73.323, -73.646, -74.109, -74.164, -73.994, -74.235,
0709        . -74.368, -73.876, -73.781, -73.797, -73.994, -73.975, -73.975,  44.622,
0710        .  44.140,  43.745,  43.745,  43.391,  42.904,  42.506,  42.115,  41.981,
0711        .  41.981,  41.509,  41.281,  41.082,  40.988,  40.618,  40.335,  40.639,
0712        .  40.733,  40.733,  40.907,  40.856,  40.726,  40.245,  40.233,  40.351/
0713          data (latitude(i),i= 5121, 5200)/
0714        .  39.898,  39.881,  39.881,  39.476,  38.982,  38.547,  38.299,  37.835,
0715        .  37.835,  38.024,  37.539,  37.163,  36.676,  36.438,  36.386,  36.386,
0716        .  36.852,  36.986,  37.485,  37.829,  38.158,  38.174,  37.951,  38.378,
0717        .  38.373,  38.321,  38.299,  38.299,  38.750,  39.174,  39.584,  39.692,
0718        .  39.692,  40.124,  40.580,  41.081,  41.583,  41.849,  41.850,  42.221,
0719        .  42.502,  42.810,  42.976,  43.408,  43.522,  43.845,  44.234,  44.699,
0720        .  45.178,  44.951,  45.320,  45.596,  45.596,  45.630,  45.486,  44.994,
0721        .  44.511,  44.080,  44.082,  43.782,  43.452,  42.977,  42.524,  42.199,
0722        .  41.919,  41.934,  41.438,  41.209,  41.021,  40.739,  40.646,  40.646,
0723        .  40.292,  39.793,  40.174,  40.365,  40.181,  39.700,  39.431,  38.932,
0724          data (latitude(i),i= 5201, 5280)/
0725        .  38.734,  38.270,  37.918,  37.918,  38.410,  38.830,  39.311,  39.776,
0726        .  40.074,  40.239,  40.239,  40.626,  40.959,  41.246,  41.415,  41.722,
0727        .  42.071,  42.089,  42.089,  42.418,  42.780,  43.242,  43.711,  44.107,
```

```
0728      .  44.347,  44.332,  43.926,  43.782,  43.782,  43.505,  43.170,  43.144,
0729      .  43.340,  43.460,  43.461,  43.157,  42.669,  42.435,  42.435,  41.934,
0730      .  41.635,  41.337,  41.188,  40.992,  40.577,  40.416,  40.416,  40.021,
0731      .  39.587,  39.086,  38.636,  38.358,  37.904,  37.606,  37.606,  37.555,
0732      .  37.215,  36.769,  36.692,  36.746,  36.732,  36.671,  36.671,  36.467,
0733      .  36.111,  36.213,  36.643,  37.080,  37.197,  37.192,  37.191,  36.999,
0734      .  37.121,  37.590,  38.093,  38.500,  38.971,  38.722,  38.722,  39.214/
0735      data (latitude(i),i= 5281, 5360)/
0736      .  39.670,  40.117,  40.618,  41.115,  41.602,  41.964,  41.964,  42.455,
0737      .  42.921,  43.303,  43.627,  43.776,  43.776,  43.596,  43.579,  43.652,
0738      .  43.514,  43.422,  43.401,  43.401,  43.508,  43.372,  43.360,  43.395,
0739      .  43.396,  43.823,  44.317,  44.818,  45.317,  45.008,  45.486,  45.707,
0740      .  45.707,  46.208,  46.508,  46.977,  47.021,  47.021,  47.373,  47.625,
0741      .  47.820,  48.004,  48.041,  48.040,  48.543,  48.713,  48.836,  48.518,
0742      .  48.692,  48.693,  48.640,  49.141,  49.610,  49.352,  49.341,  49.275,
0743      .  49.275,  49.481,  49.902,  50.210,  50.709,  51.028,  51.091,  51.091,
0744      .  51.346,  51.402,  51.904,  51.988,  51.988,  52.382,  52.871,  53.145,
0745      .  53.384,  51.651,  51.741,  51.662,  51.826,  51.686,  51.786,  53.958/
0746      data (latitude(i),i= 5361, 5440)/
0747      .  54.394,  54.669,  54.686,  55.151,  55.614,  56.118,  56.583,  56.617,
0748      .  56.169,  55.874,  55.432,  55.356,  56.583,  56.805,  57.052,  56.999,
0749      .  57.101,  57.101,  57.190,  57.571,  57.571,  57.230,  57.230,  57.078,
0750      .  57.078,  56.617,  55.356,  54.859,  54.669,  55.611,  55.120,  55.102,
0751      .  55.272,  55.767,  55.718,  55.916,  56.179,  55.734,  55.417,  55.421,
0752      .  54.686,  54.184,  53.814,  53.796,  53.700,  53.418,  53.414,  53.384,
0753      .  54.518,  54.528,  53.929,  54.171,  54.171,  54.423,  54.231,  53.941,
0754      .  53.958,  54.461,  54.343,  54.343,  54.721,  54.767,  54.570,  54.256,
0755      .  54.140,  53.947,  53.929,  65.816,  65.556,  65.059,  64.711,  64.280,
0756      .  63.887,  63.519,  63.416,  63.416,  63.094,  62.643,  62.154,  61.659/
0757      data (latitude(i),i= 5441, 5520)/
0758      .  61.160,  61.057,  61.057,  60.563,  60.391,  60.045,  59.926,  60.561,
0759      .  60.456,  60.396,  60.163,  60.004,  59.827,  59.927,  58.986,  58.492,
0760      .  58.079,  57.583,  57.114,  56.673,  56.179,  55.421,  55.857,  56.163,
0761      .  56.080,  56.538,  57.022,  57.512,  57.740,  57.740,  58.239,  58.627,
0762      .  58.731,  59.152,  59.303,  59.287,  59.478,  59.455,  59.581,  60.074,
0763      .  60.434,  60.691,  61.194,  61.313,  61.311,  61.810,  62.311,  62.759,
0764      .  63.171,  63.468,  63.602,  63.602,  63.955,  64.379,  64.848,  65.341,
0765      .  65.567,  65.806,  65.817,  69.790,  70.001,  70.257,  70.711,  70.756,
0766      .  70.270,  70.763,  70.434,  70.936,  70.526,  70.077,  70.579,  70.482,
0767      .  69.991,  70.287,  70.207,  70.207,  69.796,  69.765,  69.306,  69.791/
0768      data (latitude(i),i= 5521, 5600)/
0769      .  69.649,  69.292,  68.830,  68.481,  68.389,  68.391,  67.897,  67.874,
0770      .  67.783,  67.783,  67.386,  67.244,  67.204,  67.151,  66.756,  66.275,
0771      .  65.836,  65.332,  65.001,  64.514,  64.428,  64.428,  63.986,  63.822,
0772      .  63.462,  63.391,  63.074,  63.001,  62.723,  62.604,  62.103,  62.272,
0773      .  62.187,  62.187,  61.877,  61.682,  61.195,  61.133,  61.158,  61.044,
0774      .  61.054,  60.553,  60.100,  60.439,  60.003,  59.549,  59.564,  59.128,
0775      .  58.750,  58.750,  58.364,  58.124,  58.071,  58.359,  58.733,  59.015,
0776      .  59.502,  59.154,  58.987,  53.071,  52.927,  40.493,  40.493,  40.796,
0777      .  40.796,  40.005,  40.005,  39.205,  39.205,  38.978,  38.978,  39.039,
0778      .  38.704,  38.334,  38.452,  38.837,  39.039,  37.680,  37.680,  37.471/
0779      data (latitude(i),i= 5601, 5680)/
0780      .  37.471,  37.990,  37.990,  37.676,  37.676,  37.503,  37.503,  37.501,
0781      .  37.501,  37.174,  37.174,  36.788,  36.788,  36.477,  36.477,  35.311,
0782      .  35.007,  34.954,  35.095,  35.193,  35.523,  35.587,  35.418,  35.335,
0783      .  35.183,  35.311,  36.377,  36.377,  36.751,  36.751,  37.118,  37.118,
0784      .  37.043,  37.043,  37.999,  37.999,  37.769,  37.769,  37.923,  37.923,
0785      .  38.468,  38.468,  38.835,  38.835,  39.818,  39.818,  42.784,  42.784,
0786      .  42.778,  42.778,  42.985,  42.985,  43.196,  43.137,  43.188,  43.196,
0787      .  43.389,  43.389,  43.071,  43.071,  44.169,  44.169,  44.694,  44.344,
0788      .  44.691,  44.694,  45.165,  44.676,  45.157,  45.165,  45.214,  45.214,
0789      .  54.334,  54.334,  54.462,  54.462,  54.749,  54.749,  53.744,  53.744,
0790      data (latitude(i),i= 5681, 5760)/
0791      .  53.718,  53.718,  53.602,  53.602,  53.498,  53.498,  36.766,  36.766,
```

141

```
0792    .  38.256,   37.836,   37.358,   36.856,   36.783,   37.108,   37.298,   37.573,
0793    .  37.944,   38.146,   37.993,   38.012,   38.188,   38.297,   38.256,   42.844,
0794    .  42.844,   41.243,   40.859,   40.359,   39.857,   39.357,   39.145,   38.960,
0795    .  39.463,   39.964,   40.467,   40.947,   40.965,   41.243,   36.025,   36.025,
0796    .  35.803,   35.803,   37.001,   37.001,   37.902,   37.739,   37.902,   38.723,
0797    .  38.723,   38.527,   38.527,   39.519,   39.519,   40.030,   40.030,   39.966,
0798    .  39.472,   39.568,   39.942,   39.966,   39.123,   39.123,   38.759,   38.759,
0799    .  37.766,   37.766,   36.998,   36.998,   43.006,   42.509,   42.010,   41.526,
0800    .  41.743,   42.236,   42.634,   43.008,   43.006,   45.819,   45.819,   46.157/
0801         data (latitude(i),i= 5761, 5840)/
0802    .  46.157,   46.701,   46.701,   46.913,   46.913,   47.387,   47.387,   48.478,
0803    .  48.478,   49.254,   49.254,   49.500,   49.500,   52.564,   52.564,   53.182,
0804    .  53.182,   53.443,   53.443,   53.466,   53.466,   55.354,   55.232,   54.867,
0805    .  54.411,   54.041,   53.540,   53.488,   52.612,   53.081,   53.378,   53.821,
0806    .  54.227,   54.292,   54.471,   54.972,   55.253,   55.354,   54.010,   54.010,
0807    .  57.961,   57.685,   57.684,   57.191,   56.733,   56.415,   56.360,   56.360,
0808    .  56.051,   55.730,   55.275,   54.796,   54.675,   54.674,   54.395,   53.926,
0809    .  53.685,   53.684,   53.308,   52.810,   52.954,   52.689,   52.191,   51.784,
0810    .  51.506,   51.375,   51.347,   51.346,   50.942,   50.738,   50.797,   50.861,
0811    .  50.701,   50.729,   50.453,   50.362,   50.145,   50.134,   51.014,   51.241/
0812         data (latitude(i),i= 5841, 5920)/
0813    .  51.351,   51.733,   51.469,   51.609,   51.648,   52.135,   52.525,   52.798,
0814    .  52.789,   52.789,   53.171,   53.332,   53.727,   54.225,   54.451,   54.931,
0815    .  54.769,   54.770,   54.848,   55.346,   55.844,   55.489,   55.988,   56.438,
0816    .  56.813,   56.813,   56.563,   57.059,   57.561,   57.931,   57.971,   57.971,
0817    .  58.456,   58.557,   58.639,   58.176,   57.961,   53.488,   52.988,   52.503,
0818    .  52.353,   52.353,   52.154,   51.925,   51.643,   51.484,   51.602,   51.783,
0819    .  51.783,   52.143,   52.580,   52.612,   58.509,   58.012,   58.511,   58.509,
0820    .  57.684,   57.684,   57.488,   57.488,   57.406,   57.406,   57.049,   57.049,
0821    .  57.695,   57.248,   57.428,   57.695,   57.054,   57.054,   56.688,   56.688,
0822    .  56.534,   56.534,   56.111,   56.111,   56.142,   56.142,   55.917,   55.917/
0823         data (latitude(i),i= 5921, 6000)/
0824    .  56.650,   56.650,   51.369,   51.369,   50.134,   50.599,   51.014,   58.936,
0825    .  58.936,   58.850,   58.850,   50.678,   50.678,   49.934,   49.934,   53.425,
0826    .  53.425,   54.409,   54.409,   55.717,   55.717,   57.814,   57.814,   60.837,
0827    .  60.837,   60.466,   59.971,   60.466,   60.725,   60.725,   54.957,   54.957,
0828    .  55.272,   54.806,   55.191,   55.621,   55.964,   55.916,   55.102,   55.227,
0829    .  55.532,   55.611,   55.970,   55.970,   57.087,   57.087,   55.300,   55.300,
0830    .  55.059,   55.059,   55.148,   55.148,   55.084,   55.084,   57.303,   57.303,
0831    .  56.974,   56.974,   70.924,   70.924,   70.522,   70.522,   70.274,   70.274,
0832    .  70.478,   70.784,   70.467,   70.478,   70.246,   70.246,   69.801,   69.801,
0833    .  69.530,   69.530,   70.059,   70.059,   70.006,   70.006,   70.053,   70.053/
0834         data (latitude(i),i= 6001, 6080)/
0835    .  69.011,   69.501,   69.007,   69.011,   68.911,   68.911,   68.263,   68.723,
0836    .  68.257,   68.263,   68.617,   68.617,   68.167,   68.167,   68.044,   68.044,
0837    .  68.071,   68.071,   65.366,   65.366,   64.963,   64.963,   63.717,   63.717,
0838    .  63.551,   63.551,   63.386,   63.386,   62.334,   62.334,   62.264,   62.264,
0839    .  61.846,   61.846,   60.674,   60.674,   59.962,   59.962,   59.409,   59.409,
0840    .  58.282,   58.282,   57.338,   56.856,   56.377,   56.879,   57.337,   57.338,
0841    .  57.899,   57.401,   57.017,   57.507,   57.909,   57.899,   57.950,   57.950,
0842    .  58.390,   58.390,   64.973,   64.973,   63.244,   63.244,   60.288,   60.288,
0843    .  60.400,   60.400,   60.156,   60.156,   60.192,   60.192,   60.064,   60.064,
0844    .  66.382,   65.883,   65.523,   65.029,   64.666,   64.383,   64.383,   64.782,/
0845         data (latitude(i),i= 6081, 6160)/
0846    .  63.857,   63.746,   63.409,   63.522,   63.757,   63.849,   63.849,   63.807,
0847    .  64.187,   64.600,   64.790,   64.890,   64.889,   64.998,   65.141,   65.535,
0848    .  65.444,   65.946,   35.910,   66.328,   66.219,   65.957,   65.956,   65.454,
0849    .  65.645,   65.990,   65.866,   66.172,   66.483,   66.271,   66.384,   61.564,
0850    .  61.564,   61.873,   61.873,   62.317,   62.317,   62.357,   62.357,   62.301,
0851    .  62.301,   62.144,   62.144,   54.286,   54.619,   54.278,   54.286,   52.927,
0852    .  52.424,   52.584,   53.005,   53.071,   31.239,   31.411,   31.411,   31.542,
0853    .  31.441,   31.441,   31.292,   31.237,   31.237,   31.095,   30.842,   30.826,
0854    .  27.533,   27.099,   26.614,   26.164,   25.736,   25.278,   24.808,   24.377,
0855    .  24.013,   23.527,   23.041,   22.686,   22.341,   22.295,   30.826,   31.049/
```

142

```
0856          data (latitude(i),i= 6161, 6240)/
0857      .  31.098,  31.278,  31.445,  31.551,  31.570,  31.644,  27.535,  28.021,
0858      .  28.398,  28.802,  28.802,  29.285,  29.757,  29.951,  31.644,  32.019,
0859      .  32.000,  32.180,  32.427,  32.757,  32.912,  32.826,  32.708,  32.441,
0860      .  32.069,  31.568,  31.210,  31.210,  30.737,  30.394,  30.273,  30.549,
0861      .  30.838,  31.026,  31.188,  31.230,  31.388,  31.809,  32.296,  32.449,
0862      .  32.701,  32.789,  32.849,  32.824,  33.061,  33.171,  33.171,  33.534,
0863      .  33.648,  33.972,  34.416,  34.765,  35.196,  35.265,  35.266,  35.712,
0864      .  36.117,  36.519,  36.980,  36.764,  37.206,  37.298,  37.120,  36.942,
0865      .  36.942,  36.869,  37.052,  37.051,  36.893,  37.028,  36.825,  36.700,
0866      .  36.887,  36.917,  36.787,  36.643,  36.627,  36.628,  36.549,  36.485/
0867          data (latitude(i),i= 6241, 6320)/
0868      .  36.260,  35.917,  35.769,  35.613,  35.260,  35.093,  35.095,  35.095,
0869      .  35.183,  35.199,  35.239,  35.209,  35.509,  35.831,  35.781,  35.781,
0870      .  35.304,  34.831,  34.384,  33.963,  33.701,  33.478,  33.270,  32.871,
0871      .  32.480,  32.044,  32.044,  31.634,  31.149,  30.645,  30.194,  29.749,
0872      .  29.326,  28.961,  28.697,  28.325,  28.123,  27.993,  27.747,  27.665,
0873      .  24.003,  23.559,  23.120,  22.635,  22.263,  21.816,  21.320,  20.822,
0874      .  20.766,  27.665,  27.201,  26.728,  26.442,  26.144,  25.669,  25.196,
0875      .  24.697,  24.347,  24.002,  20.766,  20.579,  20.201,  19.706,  19.210,
0876      .  18.750,  18.255,  17.753,  17.265,  16.808,  16.326,  16.060,  16.061,
0877      .  15.574,  15.155,  14.821,  14.431,  13.976,  13.530,  13.443,  13.516/
0878          data (latitude(i),i= 6321, 6400)/
0879      .  13.409,  13.286,  12.851,  12.563,  12.610,  12.832,  12.553,  12.582,
0880      .  12.323,  12.323,  12.293,  12.016,  11.921,  11.462,  11.008,  10.981,
0881      .  10.482,  10.153,   9.872,   9.495,   9.040,   9.036,   9.035,   8.544,
0882      .   8.112,   7.694,   7.319,   7.143,   6.856,   6.557,   6.219,   5.998,
0883      .   5.656,   5.304,   5.003,   4.779,   4.557,   4.358,   4.353,   4.353,
0884      .   4.536,   4.741,   4.924,   5.061,   5.119,   5.177,   5.216,   5.210,
0885      .   5.303,   5.173,   5.352,   5.085,   5.085,   4.977,   4.781,   4.956,
0886      .   5.116,   5.277,   5.510,   5.726,   5.784,   6.050,   6.101,   6.101,
0887      .   6.234,   6.307,   6.363,   6.389,   6.577,   6.396,   6.226,   5.875,
0888      .   5.648,   5.151,   4.656,   4.370,   4.323,   4.759,   4.468,   4.515/
0889          data (latitude(i),i= 6401, 6480)/
0890      .   4.519,   4.518,   4.543,   4.545,   4.214,   4.214,   3.901,   3.631,
0891      .   3.144,   2.649,   2.348,   2.348,   1.850,   1.424,   1.076,   1.000,
0892      .   1.000,   0.535,   0.185,   0.095,  -0.372,  -0.686,  -0.686,  -1.170,
0893      .  -1.571,  -1.609,  -2.327,  -2.530,  -2.968,  -3.331,  -3.703,  -3.944,
0894      .  -1.613,  -1.389,  -1.846,  -2.276,  -2.326,  -3.946,  -4.304,  -4.708,
0895      .  -5.019,  -5.858,  -5.970,  -5.770,  -5.266,  -5.019, -12.772, -13.209,
0896      . -13.665, -14.139, -14.637, -15.114, -15.609, -16.070, -16.570, -17.072,
0897      . -17.207,  -5.858,  -6.011,  -6.448,  -6.861,  -7.333,  -7.803,  -8.261,
0898      .  -8.762,  -9.096,  -9.336,  -9.337,  -9.821, -10.272, -10.697, -11.198,
0899      . -11.700, -12.185, -12.599, -12.773, -17.207, -17.709, -18.187, -18.615/
0900          data (latitude(i),i= 6481, 6560)/
0901      . -19.021, -19.464, -19.909, -20.357, -20.837, -21.259, -21.694, -22.100,
0902      . -22.548, -22.682, -22.682, -23.138, -23.138, -23.637, -24.137, -24.616,
0903      . -25.081, -25.582, -26.076, -26.556, -27.055, -27.522, -27.970, -28.327,
0904      . -28.577, -28.577, -29.027, -29.484, -29.965, -30.439, -30.892, -31.327,
0905      . -31.742, -32.236, -32.722, -33.214, -33.640, -34.138, -34.348, -34.348,
0906      . -34.334, -34.692, -34.698, -34.441, -34.428, -34.351, -34.063, -34.085,
0907      . -33.992, -34.063, -34.187, -33.998, -34.025, -34.026, -33.739, -33.639,
0908      . -33.386, -33.090, -32.759, -32.421, -32.056, -31.668, -31.355, -30.962,
0909      . -30.534, -30.086, -29.882, -29.880, -29.423, -29.043, -28.753, -28.330,
0910      . -27.844, -27.358, -26.871, -26.847, -26.849, -26.347, -25.979, -25.565/
0911          data (latitude(i),i= 6561, 6640)/
0912      . -25.262, -25.080, -24.906, -24.856, -24.855, -24.661, -24.290, -23.813,
0913      . -23.312, -22.809, -22.305, -21.853, -21.378, -20.969, -20.969, -20.582,
0914      . -20.081, -19.604, -19.720, -19.368, -19.002, -18.863, -14.879, -14.380,
0915      . -13.877, -13.375, -12.884, -12.388, -11.888, -11.388, -10.892, -10.478,
0916      . -18.862, -18.473, -18.099, -17.701, -17.463, -17.246, -17.065, -16.831,
0917      . -16.568, -16.199, -15.776, -15.372, -14.867, -14.879, -10.479, -10.145,
0918      .  -9.752,  -9.256,  -8.782,  -8.304,  -7.826,  -7.329,  -6.826,  -6.456,
0919      .  -5.992,  -5.526,  -5.032,  -4.671,  -4.671,  -4.317,  -3.866,  -3.383,
```

```
0920        .  -2.923,   -2.535,   -2.226,   -1.958,   -1.671,   -1.674,   -1.256,   -0.833,
0921        .  -0.4C,    -0.105,    0.002,    0.001,    0.368,    0.729,    1.046,    1.389/
0922        data (latitude(i),i= 6641, 6720)/
0923        .   1.681,    1.940,    2.187,    2.450,    2.811,    3.155,    3.527,    3.884,
0924        .   3.937,    3.937,    4.317,    4.730,    5.149,    5.558,    5.988,    6.468,
0925        .   6.920,    7.313,    7.313,    7.760,    8.176,    8.629,    9.046,    9.506,
0926        .  10.003,   10.423,   10.452,   10.452,   10.869,   11.373,   11.871,   11.948,
0927        .  11.592,   11.453,   11.328,   11.244,   11.244,   11.288,   11.126,   11.181,
0928        .  10.932,   10.713,   10.791,   10.692,   10.469,   10.416,   10.621,   10.961,
0929        .  11.384,   11.574,   12.708,   12.276,   11.853,   11.567,   11.574,   18.033,
0930        .  17.574,   17.106,   16.621,   12.708,   13.032,   13.369,   13.762,   14.068,
0931        .  14.477,   14.536,   14.536,   14.704,   14.913,   15.364,   15.534,   15.959,
0932        .  16.451,   16.618,   18.033,   18.272,   18.651,   19.001,   19.493,   19.988/
0933        data (latitude(i),i= 6721, 6800)/
0934        .  20.489,   20.987,   21.474,   21.964,   22.287,   31.138,   31.137,   31.216,
0935        .  31.217,   34.739,   34.739,   33.798,   33.798,   37.532,   37.532,   35.924,
0936        .  35.924,   29.143,   29.143,   28.500,   28.094,   28.294,   28.500,   27.939,
0937        .  27.939,   28.562,   28.364,   28.480,   28.562,   28.200,   28.200,   27.756,
0938        .  27.756,   28.529,   28.529,   30.034,   30.034,   32.822,   32.822,   33.061,
0939        .  33.061,   14.899,   14.899,   15.001,   15.001,   15.001,   15.001,   15.293,
0940        .  15.293,   16.002,   16.002,   16.793,   16.793,   16.656,   16.656,   16.803,
0941        .  16.803,   17.003,   17.003,   19.853,   19.853,   11.878,   11.878,   11.626,
0942        .  11.626,   11.601,   11.601,   11.565,   11.565,   11.251,   11.251,   11.193,
0943        .  11.193,   11.202,   11.202,   11.195,   11.195,   11.302,   11.302,    7.614/
0944        data (latitude(i),i= 6801, 6880)/
0945        .   7.614,   -1.421,   -1.421,    0.404,    0.404,    1.690,    1.690,    3.748,
0946        .   3.274,    3.751,    3.748,  -16.792,  -16.792,  -26.014,  -26.014,  -21.714,
0947        . -21.714,  -11.379,  -11.846,  -11.386,  -11.379,  -12.243,  -12.243,  -12.162,
0948        . -12.162,  -12.675,  -12.675,  -12.748,  -12.748,   -9.339,   -9.339,   -9.343,
0949        .  -9.343,   -7.978,   -7.978,   -5.939,   -6.385,   -5.887,   -5.939,   -5.438,
0950        .  -4.936,   -5.430,   -5.438,   -4.073,   -4.073,   -2.202,   -2.202,   15.605,
0951        .  15.605,  -20.998,  -20.998,   -4.281,   -4.281,   -4.551,   -4.551,   -7.115,
0952        .  -7.115,  -10.330,  -10.330,  -10.397,  -10.397,  -20.123,  -20.438,  -20.123,
0953        . -19.677,  -19.677,  -13.400,  -13.400,  -17.074,  -17.074,  -25.522,  -25.499,
0954        . -25.289,  -25.023,  -24.553,  -24.082,  -23.582,  -23.099,  -22.669,  -22.165/
0955        data (latitude(i),i= 6881, 6960)/
0956        . -21.671,  -21.274,  -20.799,  -20.388,  -20.383,  -20.383,  -19.932,  -19.429,
0957        . -18.974,  -18.480,  -17.980,  -17.479,  -17.047,  -16.588,  -16.198,  -15.956,
0958        . -15.782,  -15.712,  -15.419,  -15.456,  -14.961,    1.852,  -20.997,  -21.489,
0959        . -21.969,  -22.444,  -22.945,  -23.433,  -23.925,  -24.391,  -24.861,  -25.173,
0960        . -25.245,  -25.509,  -25.522,  -14.850,  -14.627,  -14.162,  -13.665,  -13.579,
0961        . -13.280,  -12.789,  -12.298,  -11.948,  -11.948,  -12.384,  -12.828,  -13.299,
0962        . -13.774,  -14.274,  -14.774,  -15.214,  -15.707,  -15.989,  -15.989,  -15.548,
0963        . -16.010,  -16.483,  -16.919,  -17.401,  -17.904,  -18.404,  -18.869,  -19.346,
0964        . -19.825,  -20.298,  -20.778,  -20.997,  -46.451,  -46.451,  -46.120,  -46.120,
0965        . -46.958,  -46.958,  -40.391,  -40.391,  -37.115,  -37.115,  -15.994,  -15.994/
0966        data (latitude(i),i= 6961, 7040)/
0967        .  -7.983,   -7.983,    1.009,    1.511,    1.855,    2.164,    1.662,    1.362,
0968        .   1.009,   -4.309,   -3.814,   -3.342,   -3.838,   -4.298,   -4.798,   -5.256,
0969        .  -5.745,   -5.922,   -5.921,   -6.417,   -6.630,   -7.071,   -7.570,   -7.997,
0970        .  -8.426,   -8.489,   -8.021,   -7.889,   -7.890,   -7.398,   -7.064,   -6.670,
0971        .  -6.199,   -5.697,   -5.234,   -4.735,   -4.250,   -4.307,  -17.974,  -17.566,
0972        . -17.117,  -16.791,  -16.557,  -16.559,  -16.780,  -17.068,  -17.351,  -17.709,
0973        . -17.974,   -9.700,  -10.180,  -10.666,  -11.169,  -11.664,  -12.089,  -12.586,
0974        . -13.070,  -13.563,  -14.051,  -14.175,  -13.703,  -13.200,  -12.697,  -12.206,
0975        . -11.760,  -11.566,  -11.565,  -11.156,  -10.655,  -10.165,   -9.723,   -9.700,
0976        .  -2.046,   -1.563,   -1.071,   -0.569,   -0.100,    0.045,    0.046,    0.184
0977        data (latitude(i),i= 7041, 7120)/
0978        .   0.339,    0.325,   -0.070,   -0.190,   -0.191,   -0.092,   -0.527,   -0.937,
0979        .  -1.327,   -1.762,   -2.109,   -2.112,   -2.533,   -2.557,   -2.557,   -2.393,
0980        .  -2.882,   -2.409,   -2.389,   -2.781,   -2.326,   -2.044,   11.632,   12.026,
0981        .  12.296,   11.790,   11.632,  -17.095,  -17.095,   35.926,   36.426,   36.910,
0982        .  36.587,   36.718,   36.641,   36.251,   36.138,   36.021,   36.021,   36.284,
0983        .  36.599,   36.819,   36.825,   36.327,   36.222,   36.263,   36.710,   36.697,
```

```
0984      .  36.697,   36.567,   36.662,   36.788,   37.007,   37.476,   37.978,   38.204,
0985      .  38.638,   38.373,   38.866,   39.269,   39.471,   39.470,   39.966,   40.316,
0986      .  40.464,   40.507,   40.380,   40.646,   40.719,   40.900,   41.156,   41.212,
0987      .  41.212,   41.075,   41.340,   41.582,   41.833,   42.002,   41.979,   41.940/
0988         data (latitude(i),i= 7121, 7200)/
0989      .  42.091,   42.090,   41.684,   41.655,   41.241,   41.150,   41.118,   40.920,
0990      .  41.038,   41.008,   40.976,   35.005,   35.502,   35.926,   35.005,   34.648,
0991      .  34.648,   34.224,   33.752,   33.291,   33.095,   33.095,   32.620,   32.132,
0992      .  31.673,   31.596,   31.596,   31.326,   31.326,   31.122,   31.179,   31.044,
0993      .  31.044,   31.193,   29.951,   29.466,   29.069,   28.586,   28.294,   28.294,
0994      .  27.913,   28.119,   28.542,   28.542,   29.022,   29.486,   29.552,   29.552,
0995      .  29.546,   29.546,   29.360,   29.360,   28.870,   28.378,   28.139,   28.139,
0996      .  27.964,   27.964,   27.524,   27.096,   26.693,   26.251,   25.805,   25.441,
0997      .  24.976,   24.486,   24.174,   24.174,   23.896,   23.499,   23.018,   22.583,
0998      .  22.083,   21.583,   21.091,   20.685,   20.569,   20.568,   20.274,   20.000/
0999         data (latitude(i),i= 7201, 7280)/
1000      .  19.611,   19.167,   18.688,   18.270,   17.838,   17.531,   17.050,   16.686,
1001      .  16.196,   15.706,   15.214,   14.735,   14.244,   13.773,   13.279,   13.278,
1002      .  12.837,   12.603,   12.694,   12.821,   13.057,   13.369,   13.417,   13.479,
1003      .  13.631,   13.912,   13.989,   14.098,   14.497,   14.512,   14.512,   14.750,
1004      .  14.843,   15.055,   15.165,   15.325,   15.578,   16.071,   16.448,   16.628,
1005      .  16.749,   16.977,   17.036,   17.034,   17.174,   17.174,   17.661,   17.904,
1006      .  17.937,   18.359,   18.788,   18.935,   19.309,   19.810,   20.004,   20.004,
1007      .  20.460,   20.391,   24.748,   25.181,   25.599,   26.048,   26.550,   26.964,
1008      .  27.331,   27.609,   28.078,   28.545,   28.546,   20.391,   20.844,   21.245,
1009      .  21.585,   22.033,   22.522,   22.765,   23.168,   23.566,   23.705,   23.797/
1010         data (latitude(i),i= 7281, 7360)/
1011      .  23.988,   24.372,   24.804,   25.303,   25.636,   25.636,   26.128,   26.066,
1012      .  26.066,   25.656,   25.312,   24.948,   24.546,   24.310,   24.052,   24.124,
1013      .  24.122,   23.984,   24.353,   24.617,   24.618,   24.986,   25.480,   25.972,
1014      .  25.731,   25.243,   24.747,   28.545,   28.997,   29.386,   29.864,   30.012,
1015      .  30.012,   29.923,   29.964,   29.963,   30.371,   30.165,   30.216,   29.791,
1016      .  29.409,   28.947,   28.476,   28.154,   35.672,   35.373,   34.974,   34.699,
1017      .  34.673,   35.174,   35.361,   35.390,   35.574,   35.672,   40.145,   40.145,
1018      .  39.326,   39.326,   38.538,   38.538,   37.679,   37.679,   37.771,   37.771,
1019      .  37.004,   37.004,   36.859,   36.859,   36.636,   36.636,   36.443,   35.999,
1020      .  36.443,   35.815,   17.003,   17.003,   16.876,   16.584,   16.876/
1021         data (latitude(i),i= 7361, 7440)/
1022      .  12.240,   12.240,   12.651,   12.321,   12.504,   12.624,   12.651,   17.558,
1023      .  17.558,   20.692,   20.218,   20.693,   20.692,   24.230,   24.230,   24.545,
1024      .  24.545,   24.589,   24.589,   24.326,   24.326,   26.240,   26.240,   29.983,
1025      .  29.983,   30.778,   30.778,   30.758,   30.758,   30.900,   30.900,   30.753,
1026      .  30.752,   30.823,   30.823,   30.811,   30.811,   30.593,   30.593,   30.419,
1027      .  30.419,   30.678,   30.745,   30.701,   30.678,   31.194,   30.812,   30.812,
1028      .  30.588,   30.564,   30.420,   30.176,   29.952,  -37.895,  -37.895,  -38.722,
1029      . -38.722,  -49.719,  -49.561,  -49.228,  -48.738,  -48.983,  -49.055,  -49.410,
1030      . -49.719,  -49.083,  -49.083,  -52.988,  -52.988,  -54.459,  -54.459,   53.003,
1031      .  53.003,   52.446,   52.446,   52.119,   52.119,   51.649,   51.408,   51.618/
1032         data (latitude(i),i= 7441, 7520)/
1033      .  51.649,   51.829,   51.829,   51.909,   51.909,   51.889,   51.889,   51.972,
1034      .  51.972,   51.857,   51.857,   51.970,   31.876,   51.909,   51.909,   52.095,
1035      .  52.095,   51.987,   51.987,   51.970,   51.970,   52.419,   52.048,   52.419,
1036      .  52.123,   52.106,   52.095,   52.123,   52.382,   52.382,   52.524,   52.524,
1037      .  52.684,   52.684,   52.778,   52.778,   53.035,   53.035,   52.855,   52.855,
1038      .  53.496,   53.131,   52.833,   53.267,   53.518,   53.496,   53.967,   53.525,
1039      .  53.324,   53.676,   53.967,   54.117,   54.117,   54.087,   54.087,   54.125,
1040      .  54.125,   54.292,   54.292,   56.613,   56.613,   57.209,   57.209,   60.410,
1041      .  59.976,   59.852,   60.161,   60.380,   60.410,   60.586,   60.586,   62.943,
1042      .  63.330,   63.327,   63.667,   63.436,   62.943,   65.749,   65.749,   52.022
1043         data (latitude(i),i= 7521, 7600)/
1044      .  52.022,   20.272,   20.060,   19.692,   19.279,   18.927,   19.381,   19.881,
1045      .  20.272,   20.939,   20.939,   20.606,   20.606,   20.925,   20.925,   21.203,
1046      .  21.203,   21.697,   21.697,   22.183,   22.183,   22.009,   22.009,   21.641,
1047      .  21.641,   23.043,   23.043,   23.578,   23.578,   25.011,   25.011,   25.788,
```

145

```
1048    . 25.788,  26.078,  26.078,  28.209,  28.209,  28.219,  28.219,  28.391,
1049    . 28.391,  16.729,  16.729,  44.623,  44.814,  45.211,  71.141,  71.141,
1050    . 74.512,  74.512,  76.684,  76.684,  78.001,  77.533,  77.487,  77.984,
1051    . 78.001,  78.243,  78.243,  78.804,  78.804,  78.960,  78.960,  80.342,
1052    . 80.036,  79.592,  79.206,  79.381,  79.711,  80.129,  80.421,  80.038,
1053    . 80.299,  80.342,  80.221,  80.221,  79.538,  79.081,  78.589,  78.201/
1054      data (latitude(i),i= 7601, 7680)/
1055    . 78.683,  78.409,  78.104,  77.926,  77.676,  77.186,  77.001,  77.440,
1056    . 77.916,  78.400,  78.668,  79.095,  79.551,  79.959,  79.502,  79.010,
1057    . 79.477,  79.549,  79.538,  54.286,  54.619,  54.278,  54.286,  27.140,
1058    . 26.695,  26.205,  25.756,  25.679,  25.593,  25.396,  25.398,  25.332,
1059    . 25.262,  25.117,  25.197,  25.198,  25.216,  25.259,  25.230,  25.373,
1060    . 25.281,  25.323,  25.383,  25.423,  25.593,  25.593,  25.172,  24.769,
1061    . 24.343,  24.064,  24.064,  23.928,  23.928,  23.910,  23.910,  23.937,
1062    . 23.938,  23.774,  23.774,  23.844,  23.430,  23.064,  22.825,  22.854,
1063    . 22.936,  22.550,  22.373,  22.274,  22.275,  21.803,  21.464,  21.072,
1064    . 20.787,  20.760,  20.968,  21.126,  21.127,  21.523,  22.001,  22.277/
1065      data (latitude(i),i= 7681, 7760)/
1066    . 21.772,  21.747,  21.494,  21.033,  20.529,  20.057,  19.554,  19.217,
1067    . 19.217,  18.750,  18.273,  17.780,  17.295,  16.798,  16.303,  15.854,
1068    . 15.428,  14.945,  14.518,  14.244,  14.244,  13.754,  13.261,  12.771,
1069    . 12.313,  11.921,  11.469,  11.016,  10.543,  10.070,   9.666,   9.676,
1070    .  9.658,   9.675,   9.666,   9.658,   9.173,   8.744,   8.354,   8.087,
1071    .  8.288,   8.656,   8.655,   9.082,   9.247,   9.199,   9.435,   9.887,
1072    . 10.152,  10.152,  10.261,  10.752,  11.255,  11.753,  12.214,  12.662,
1073    . 13.156,  13.593,  14.088,  14.562,  14.562,  15.054,  15.531,  15.875,
1074    . 15.941,  16.369,  16.386,  16.680,  17.169,  17.412,  17.708,  18.083,
1075    . 18.333,  18.332,  18.708,  19.144,  19.454,  19.883,  19.820,  20.044/
1076      data (latitude(i),i= 7761, 7840)/
1077    . 20.407,  20.901,  21.400,  21.611,  21.829,  22.329,  22.235,  22.236,
1078    . 21.744,  22.062,  21.566,  21.918,  21.918,  22.229,  21.729,  22.133,
1079    . 22.471,  22.614,  22.128,  22.086,  22.584,  23.088,  23.460,  23.460,
1080    . 23.100,  22.655,  22.753,  22.365,  21.883,  21.388,  20.930,  21.055,
1081    . 21.055,  20.605,  20.180,  19.816,  19.910,  19.459,  19.014,  19.364,
1082    . 19.385,  19.198,  19.255,  19.385,  18.882,  18.447,  17.964,  17.472,
1083    . 16.982,  16.514,  16.016,  16.338,  16.161,  16.160,  15.788,  15.928,
1084    . 16.267,  16.746,  16.619,  17.077,  17.451,  17.451,  17.101,  16.606,
1085    . 16.187,  15.691,  15.198,  14.726,  14.228,  13.727,  13.382,  13.319,
1086    . 13.320,  12.843,  12.347,  11.846,  11.347,  10.863,  10.365,  10.349/
1087      data (latitude(i),i= 7841, 7920)/
1088    . 10.349,   9.878,   9.402,   8.911,   8.412,   8.225,   7.868,   7.454,
1089    .  7.130,   6.664,   6.419,   6.237,   6.576,   6.924,   6.959,   7.199,
1090    .  7.634,   7.255,   7.749,   8.242,   8.631,   9.129,   9.190,   9.189,
1091    .  9.189,   9.653,  10.156,  10.639,  11.109,  11.577,  12.024,  12.521,
1092    . 13.015,  13.364,  13.372,  13.500,  13.238,  12.745,  12.657,  12.657,
1093    . 12.705,  12.388,  12.065,  11.709,  11.641,   6.420,   5.969,   5.468,
1094    .  4.969,   4.511,   4.031,   3.609,   3.183,   2.708,   2.407,   2.158,
1095    .  2.039,   2.039,   1.737,   1.499,   1.435,   1.886,   2.348,   2.717,
1096    .  3.200,   3.505,   3.504,   4.004,   4.507,   4.994,   5.429,   5.725,
1097    .  6.132,   6.237,  11.641,  11.174,  11.086,  10.589,  10.557,  10.423/
1098      data (latitude(i),i= 7921, 8000)/
1099    . 10.423,  10.129,   9.626,   9.187,   9.187,   8.709,   8.961,   9.234,
1100    .  9.412,   9.412,   9.870,   9.565,  10.058,  10.088,  10.482,  10.481,
1101    . 10.489,  10.576,  10.807,  11.060,  11.314,  11.715,  12.215,  12.714,
1102    . 13.079,  13.078,  13.575,  14.078,  14.559,  15.037,  15.486,  15.893,
1103    . 16.284,  16.539,  16.888,  17.013,  17.012,  17.360,  17.740,  18.218,
1104    . 18.554,  19.000,  19.467,  19.942,  20.203,  20.302,  20.302,  20.724,
1105    . 20.995,  21.336,  21.545,  22.510,  22.549,  21.545,  21.607,  21.642,
1106    . 21.744,  21.452,  21.453,  21.015,  20.527,  20.257,  20.756,  21.251,
1107    . 21.401,  21.481,  21.481,  21.616,  21.837,  21.889,  22.049,  22.547,
1108    . 22.188,  22.686,  23.154,  22.662,  22.521,  22.520,  22.511,  22.549
1109      data (latitude(i),i= 8001, 8080)/
1110    . 22.449,  22.450,  22.754,  22.686,  22.833,  23.002,  23.415,  23.710,
1111    . 24.021,  24.460,  24.461,  24.584,  24.869,  25.185,  25.568,  25.967,
```

```
1112    .  26.283,   26.776,   26.626,   27.110,   27.451,   27.924,   28.110,   28.109,
1113    .  28.233,   28.591,   29.088,   29.523,   29.978,   30.305,   30.123,   30.195,
1114    .  30.195,   30.390,   30.650,   30.863,   31.352,   31.672,   31.992,   31.918,
1115    .  32.274,   31.943,   32.069,   31.854,   31.689,   32.116,   32.510,   32.982,
1116    .  33.440,   33.903,   34.354,   34.566,   40.097,   39.635,   39.539,   39.037,
1117    .  38.639,   38.589,   38.139,   37.899,   37.844,   37.798,   37.834,   38.619,
1118    .  38.959,   39.263,   39.750,   40.030,   40.290,   40.592,   40.863,   41.361,
1119    .  41.858,   42.219,   42.291,   37.834,   37.331,   36.844,   36.866,   36.430/
1120       data (latitude(i),i= 8081, 8160)/
1121    .  36.061,   36.058,   35.652,   35.157,   34.674,   34.684,   34.776,   34.949,
1122    .  35.117,   35.116,   35.117,   35.455,   35.934,   36.429,   36.932,   37.410,
1123    .  37.840,   38.272,   38.619,   42.289,   42.596,   43.037,   43.219,   42.853,
1124    .  42.757,   42.935,   43.231,   43.503,   43.947,   44.364,   44.742,   45.120,
1125    .  45.491,   45.877,   46.303,   46.764,   47.198,   47.630,   48.035,   48.415,
1126    .  48.566,   48.567,   49.056,   49.543,   50.041,   50.527,   51.010,   51.493,
1127    .  51.907,   52.392,   52.893,   53.254,   53.254,   53.569,   53.984,   54.227,
1128    .  54.296,   53.792,   53.577,   54.014,   53.798,   54.297,   54.599,   54.739,
1129    .  55.151,   55.524,   55.811,   56.156,   56.600,   56.996,   57.318,   57.703,
1130    .  58.097,   58.485,   58.865,   59.229,   59.341,   59.375,   59.410,   59.456/
1131       data (latitude(i),i= 8161, 8240)/
1132    .  59.285,   59.396,   59.688,   59.652,   59.582,   59.271,   58.903,   59.011,
1133    .  59.092,   59.259,   59.267,   59.137,   59.534,   59.990,   60.408,   60.718,
1134    .  61.167,   61.589,   61.773,   61.903,   61.847,   61.344,   60.866,   61.054,
1135    .  61.429,   61.714,   62.198,   62.609,   62.567,   62.450,   62.452,   62.463,
1136    .  62.010,   61.524,   61.042,   60.678,   60.423,   59.988,   59.599,   59.244,
1137    .  58.804,   58.408,   58.072,   57.817,   57.402,   56.975,   56.575,   56.084,
1138    .  55.584,   55.079,   54.582,   54.091,   53.596,   53.098,   52.603,   52.118,
1139    .  51.618,   51.132,   51.352,   51.717,   52.154,   52.652,   52.651,   53.044,
1140    .  53.240,   53.740,   54.215,   54.538,   54.516,   54.971,   55.441,   55.921,
1141    .  56.328,   56.641,   57.095,   57.569,   57.802,   58.288,   58.733,   59.162/
1142       data (latitude(i),i= 8241, 8320)/
1143    .  59.629,   59.824,   59.824,   59.987,   60.132,   60.433,   59.932,   60.245,
1144    .  60.537,   60.567,   60.381,   59.941,   60.394,   60.714,   61.066,   61.458,
1145    .  61.701,   61.884,   62.174,   62.434,   62.863,   62.866,   62.561,   62.371,
1146    .  62.741,   63.222,   63.519,   64.023,   64.372,   64.663,   64.746,   64.845,
1147    .  64.704,   64.704,   65.030,   65.072,   65.505,   65.973,   66.410,   65.927,
1148    .  65.478,   65.599,   65.437,   64.948,   64.733,   64.386,   64.619,   64.616,
1149    .  64.425,   64.846,   65.267,   65.514,   65.898,   66.399,   66.781,   67.001,
1150    .  67.086,   66.603,   67.030,   67.517,   67.895,   68.184,   68.522,   68.821,
1151    .  68.982,   69.296,   69.538,   69.771,   69.852,   69.839,   69.842,   69.971,
1152    .  70.094,   69.666,   69.240,   68.749,   69.011,   69.285,   69.768,   69.522/
1153       data (latitude(i),i= 8321, 8400)/
1154    .  69.597,   69.691,   69.646,   69.241,   68.743,   69.236,   69.678,   70.133,
1155    .  70.637,   70.914,   71.095,   70.961,   70.915,   70.789,   70.989,   70.987,
1156    .  71.394,   71.669,   72.173,   72.329,   72.026,   71.677,   72.171,   72.269,
1157    .  72.345,   72.636,   72.702,   72.863,   72.475,   71.983,   71.490,   71.548,
1158    .  71.472,   71.513,   71.515,   71.505,   71.507,   71.289,   70.817,   71.080,
1159    .  71.506,   72.008,   72.309,   71.832,   72.279,   72.542,   72.726,   72.920,
1160    .  72.999,   73.181,   73.581,   73.695,   73.580,   73.091,   72.607,   72.595,
1161    .  72.595,   73.071,   73.569,   73.765,   74.015,   73.508,   73.307,   73.144,
1162    .  72.802,   73.281,   73.622,   73.948,   74.345,   74.680,   74.996,   75.459,
1163    .  75.836,   76.328,   76.700,   76.734,   76.518,   77.018,   77.084,   77.551/
1164       data (latitude(i),i= 8401, 8480)/
1165    .  77.630,   77.291,   76.840,   76.419,   76.479,   75.975,   75.474,   75.166,
1166    .  75.170,   75.592,   76.078,   76.057,   75.861,   76.115,   75.758,   75.608,
1167    .  75.350,   75.151,   74.645,   74.146,   73.670,   73.383,   72.994,   72.991,
1168    .  73.406,   73.778,   73.699,   73.658,   73.641,   73.339,   72.859,   72.364,
1169    .  72.040,   71.691,   71.203,   70.720,   70.254,   70.086,   70.086,   70.589,
1170    .  71.007,   71.433,   71.702,   72.104,   72.381,   72.210,   71.979,   71.492,
1171    .  71.275,   71.070,   71.070,   71.302,   71.795,   72.269,   72.765,   72.274,
1172    .  71.909,   71.482,   71.022,   70.556,   70.092,   69.593,   69.091,   69.189,
1173    .  69.136,   68.796,   68.314,   67.871,   67.601,   67.567,   67.567,   67.555,
1174    .  68.020,   68.523,   68.956,   68.867,   68.466,   67.980,   67.513,   67.039
1175       data (latitude(i),i= 8481, 8560)/
```

```
1176    .  66.682,   66.281,   66.367,   66.474,   66.870,   66.870,   66.520,   66.973,
1177    .  67.312,   67.724,   68.228,   68.729,   69.176,   69.680,   70.184,   70.687,
1178    .  71.181,   71.676,   72.158,   72.662,   72.909,   72.908,   72.888,   72.476,
1179    .  71.986,   71.521,   71.177,   70.705,   70.205,   69.714,   69.509,   69.020,
1180    .  68.538,   68.312,   68.312,   68.609,   68.935,   69.206,   69.533,   69.717,
1181    .  69.794,   69.478,   69.035,   68.707,   68.974,   68.696,   68.582,   68.581,
1182    .  68.486,   68.219,   68.721,   68.665,   68.396,   68.142,   67.855,   67.618,
1183    .  67.118,   66.823,   66.824,   66.825,   67.239,   67.739,   67.813,   68.273,
1184    .  68.578,   68.632,   68.168,   67.666,   67.177,   66.728,   66.242,   66.411,
1185    .  66.234,   66.167,   66.167,   65.902,   65.543,   65.065,   64.588,   64.627/
1186         data (latitude(i),i= 8561, 8640)/
1187    .  64.718,   65.124,   64.831,   64.409,   64.106,   63.924,   63.924,   63.967,
1188    .  64.346,   64.684,   65.174,   65.680,   66.160,   66.480,   66.872,   67.151,
1189    .  67.150,   66.912,   66.725,   66.461,   66.297,   66.129,   66.078,   66.266,
1190    .  66.641,   67.101,   67.586,   67.958,   68.059,   68.059,   68.369,   68.731,
1191    .  69.035,   69.271,   69.309,   69.457,   69.947,   69.790,   60.561,   60.337,
1192    .  60.189,   59.847,   59.763,   59.445,   59.577,   59.490,   59.490,   59.473,
1193    .  59.219,   58.730,   58.247,   57.753,   57.737,   57.743,   57.243,   56.973,
1194    .  57.377,   57.700,   57.429,   56.954,   56.467,   55.965,   55.473,   54.971,
1195    .  55.467,   55.042,   54.741,   54.462,   40.729,   40.891,   40.901,   38.289,
1196    .  38.122,   37.955,   37.451,   37.123,   37.151,   40.731,   40.228,   39.879/
1197         data (latitude(i),i= 8641, 8720)/
1198    .  39.386,   38.934,   38.943,   39.324,   39.594,   39.823,   39.833,   40.096,
1199    .  38.289,   38.764,   39.199,   39.155,   39.329,   39.782,   40.052,   40.327,
1200    .  40.699,   40.901,   34.566,   34.943,   35.363,   35.753,   36.212,   36.433,
1201    .  36.697,   36.898,   36.922,   37.395,   37.544,   37.483,   37.771,   37.683,
1202    .  37.340,   37.151,   45.212,   45.714,   45.317,   45.580,   45.817,   45.817,
1203    .  46.206,   46.607,   46.625,   46.625,   46.720,   47.209,   46.731,   46.237,
1204    .  46.129,   46.041,   45.683,   45.194,   44.760,   44.395,   44.743,   44.908,
1205    .  44.991,   45.284,   45.385,   45.592,   46.055,   46.499,   46.256,   46.256,
1206    .  46.611,   46.750,   46.924,   47.097,   47.238,   47.116,   46.861,   46.622,
1207    .  46.671,   46.671,   46.300,   45.814,   45.433,   45.424,   44.982,   44.637/
1208         data (latitude(i),i= 8721, 8800)/
1209    .  44.368,   44.155,   43.797,   43.431,   43.318,   43.318,   43.074,   42.760,
1210    .  42.298,   41.809,   41.388,   41.173,   40.940,   40.976,   24.073,   24.073,
1211    .  21.942,   21.942,   21.867,   21.866,   21.886,   21.886,   21.595,   21.594,
1212    .  -0.603,   -0.603,   -0.656,   -0.656,   -0.682,   -0.682,   -0.671,   -0.671,
1213    .  -0.626,   -0.626,   -0.589,   -0.589,   -0.284,   -0.284,    0.496,    0.496,
1214    .   0.403,    0.403,    0.305,    0.305,    0.270,    0.270,    0.228,    0.228,
1215    .   0.240,    0.240,    0.285,    0.285,    0.388,    0.388,    0.508,    0.508,
1216    .   0.848,    0.848,    0.801,    0.801,    0.843,    0.843,    2.033,    2.033,
1217    .   1.991,    1.991,    1.893,    1.893,    1.830,    1.830,    1.791,    1.791,
1218    .   1.806,    1.806,    1.843,    1.843,    1.933,    1.933,    1.976,    1.976/
1219         data (latitude(i),i= 8801, 8880)/
1220    .   2.027,    2.027,    2.082,    2.082,    2.110,    2.110,    2.327,    2.327,
1221    .   2.293,    2.293,    2.187,    2.187,    2.200,    2.200,    2.348,    2.348,
1222    .   2.437,    2.437,    2.506,    2.506,    2.694,    2.694,    2.682,    2.682,
1223    .   2.742,    2.742,    2.953,    2.953,    2.778,    2.778,    2.904,    2.904,
1224    .   2.893,    2.893,    2.916,    2.916,    2.955,    2.955,    2.962,    2.962,
1225    .   3.110,    3.110,    3.374,    3.374,    3.451,    3.451,    3.552,    3.552,
1226    .   3.511,    3.511,    3.517,    3.517,    3.727,    3.727,    4.003,    4.003,
1227    .   4.266,    4.266,    3.945,    3.945,    3.847,    3.847,    3.942,    3.942,
1228    .   3.973,    3.973,    4.104,    4.104,    4.173,    4.173,    4.178,    4.178,
1229    .   4.191,    4.191,    4.342,    4.342,    4.371,    4.371,    4.435,    4.435/
1230         data (latitude(i),i= 8881, 8960)/
1231    .   4.453,    4.453,    4.474,    4.474,    4.630,    4.630,    4.896,    4.896,
1232    .   4.971,    4.971,    5.133,    5.133,    5.173,    5.173,    5.299,    5.299,
1233    .   5.454,    5.454,    5.368,    5.368,    5.362,    5.362,    5.308,    5.308,
1234    .   5.283,    5.283,    5.419,    5.419,    5.981,    5.981,    5.969,    5.969,
1235    .   5.864,    5.864,    5.814,    5.814,    5.771,    5.771,    5.738,    5.738,
1236    .   5.841,    5.841,    5.911,    5.911,    5.970,    5.970,    6.157,    6.157,
1237    .   6.189,    6.189,    6.240,    6.240,    6.287,    6.287,    6.330,    6.330,
1238    .   6.327,    6.327,    6.351,    6.351,    6.442,    6.442,    6.412,    6.412,
1239    .   6.622,    6.622,    6.668,    6.668,    6.710,    6.710,    6.774,    6.774,
```

```
1240        .   6.831,    6.831,    6.919,    6.919,    6.971,    6.971,    6.963,    6.963/
1241        data (latitude(i),i= 8961, 9040)/
1242        .   7.028,    7.028,    7.093,    7.093,    3.292,    3.292,   11.694,   11.694,
1243        .  11.595,   11.595,   11.212,   11.212,   11.502,   11.502,   11.246,   11.246,
1244        .  11.119,   11.119,   10.779,   10.779,   10.567,   10.567,   10.830,   10.830,
1245        .  10.145,   10.145,   10.085,   10.085,    8.303,    8.303,   10.057,   10.057,
1246        .  10.137,   10.137,   10.949,   10.949,   10.864,   10.864,   -7.436,   -7.436,
1247        .  -6.653,   -6.653,   -6.389,   -6.389,   -6.229,   -6.229,   -6.196,   -6.196,
1248        .  -6.123,   -6.123,   -6.168,   -6.168,   -5.328,   -5.328,   -5.360,   -5.360,
1249        .  -5.235,   -5.235,   -5.251,   -5.251,   -5.237,   -5.237,   -5.285,   -5.285,
1250        .  -5.440,   -5.440,   -5.408,   -5.408,    7.600,    7.098,    6.604,    6.126,
1251        .   5.936,    6.102,    6.324,    6.717,    7.213,    7.677,    8.104,    8.252/
1252        data (latitude(i),i= 9041, 9120)/
1253        .   8.251,    8.705,    9.111,    9.499,    9.821,    9.482,    9.594,    9.094,
1254        .   8.614,    8.122,    7.622,    7.601,    7.683,    7.679,    9.102,    9.102,
1255        .   9.535,    9.535,    9.760,    9.760,   21.885,   21.884,   12.895,   12.897,
1256        .  12.307,   12.307,   12.179,   12.069,   21.995,   21.995,   22.044,   22.044,
1257        .  22.037,   22.037,   21.963,   21.963,   22.158,   22.158,   22.775,   22.302,
1258        .  22.727,   22.775,   22.446,   22.446,   22.877,   22.877,   23.034,   23.034,
1259        .  23.058,   23.058,   22.606,   22.606,   22.449,   22.449,   22.516,   22.516,
1260        .  22.355,   22.355,   22.251,   22.251,   22.197,   22.197,   21.913,   21.911,
1261        .  21.755,   21.755,   23.506,   23.506,   20.084,   20.081,   19.951,   19.951,
1262        .  19.560,   19.560,   18.890,   18.885,   14.139,   14.134,   14.902,   14.900/
1263        data (latitude(i),i= 9121, 9200)/
1264        .  16.231,   16.228,   15.933,   15.924,    7.605,    7.605,    9.411,    9.411,
1265        .   9.471,    9.471,   16.507,   16.506,   15.565,   15.561,   14.179,   14.180,
1266        .  13.939,   13.934,   13.859,   13.859,   13.444,   13.444,   12.974,   12.975,
1267        .  12.773,   12.774,   12.361,   12.368,   12.502,   12.506,   12.417,   12.423,
1268        .  12.371,   12.375,   12.289,   12.294,   12.169,   12.174,   12.544,   12.548,
1269        .  11.844,   11.846,   11.780,   11.782,   11.655,   11.661,   11.893,   11.904,
1270        .  11.608,   11.611,   11.928,   11.930,   11.858,   11.864,   11.453,   11.461,
1271        .  11.224,   11.229,   11.006,   11.006,   10.831,   10.833,   10.976,   10.978,
1272        .  10.880,   10.881,   10.441,   10.436,   10.521,   10.520,   10.163,   10.163,
1273        .   9.803,    9.797,   10.007,   10.007,   10.727,   10.725,   13.664,   13.662/
1274        data (latitude(i),i= 9201, 9280)/
1275        .  13.532,   13.034,   12.532,   12.081,   11.579,   12.069,   12.568,   13.063,
1276        .  13.544,   13.533,   12.960,   12.951,   12.144,   12.145,   12.015,   12.015,
1277        .  12.271,   12.269,   11.580,   11.576,   11.468,   11.468,   10.786,   10.777,
1278        .   9.239,    9.239,    8.323,    8.324,    8.545,    8.542,    8.214,    8.209,
1279        .   8.007,    7.999,    8.014,    8.012,    7.418,    7.413,    7.414,    7.410,
1280        .   7.190,    7.182,   13.174,   13.175,    9.762,    9.756,    9.081,    9.072,
1281        .  12.948,   12.948,    8.170,    8.170,    8.108,    8.100,    8.166,    8.162,
1282        .   7.266,    7.261,    6.571,    6.571,    7.174,    7.168,    9.550,    9.544,
1283        .   9.793,    9.788,    9.542,    9.542,    9.623,    9.620,    6.709,    6.703,
1284        .  12.298,   12.297,   12.140,   12.142,   11.753,   11.754,    6.260,    6.259/
1285        data (latitude(i),i= 9281, 9360)/
1286        .   6.464,    6.464,    5.449,    5.446,    2.871,    2.868,    1.388,    1.382,
1287        .  11.406,   11.403,   10.442,   10.439,    8.766,    8.765,    9.926,    9.926,
1288        .  10.659,   10.662,   10.553,   10.552,    7.872,    7.872,    9.708,    9.708,
1289        .   9.419,    9.419,   20.871,   20.873,   20.834,   20.836,   20.844,   20.840,
1290        .  21.034,   21.032,   20.958,   20.956,   20.136,   20.139,   21.036,   21.037,
1291        .  21.016,   21.017,   21.113,   21.118,   21.273,   21.271,   21.354,   21.358,
1292        .  21.384,   21.386,   15.790,   15.790,   16.051,   16.051,   16.666,   16.666,
1293        .  16.839,   16.839,   16.936,   16.936,   16.983,   16.983,   16.982,   16.982,
1294        .  16.463,   16.463,   16.544,   16.544,   16.447,   16.447,   20.720,   20.720,
1295        .  21.054,   21.055,   19.977,   19.966,   20.106,   19.749,   19.351,   18.883/
1296        data (latitude(i),i= 9361, 9440)/
1297        .  18.533,   18.199,   18.288,   18.496,   18.994,   19.451,   19.717,   19.983,
1298        .  21.056,   21.058,   21.196,   21.199,   24.434,   24.439,   25.425,   25.426,
1299        .  29.079,   29.086,   29.681,   29.686,   30.063,   30.067,   31.799,   31.580,
1300        .  31.782,   31.801,   37.978,   37.979,   37.811,   37.804,   34.499,   34.492,
1301        .  33.542,   33.226,   33.544,   33.536,   34.915,   34.902,   34.950,   34.943,
1302        .  37.528,   37.518,   25.119,   25.120,   24.618,   24.141,   23.651,   23.162,
1303        .  22.728,   22.286,   22.487,   22.954,   23.453,   23.931,   24.373,   24.778,
```

```
1304     .  25.122,   22.045,   22.045,   23.534,   23.536,   24.420,   24.419,   32.454,
1305     .  32.867,   33.246,   33.609,   33.933,   33.574,   33.354,   32.883,   32.837,
1306     .  32.694,   34.085,   33.620,   33.501,   33.285,   32.845,   33.106,   33.605/
1307        data (latitude(i),i= 9441, 9520)/
1308     .  34.016,   33.977,   34.326,   34.200,   34.065,   41.439,   40.976,   40.499,
1309     .  40.038,   39.556,   39.089,   38.668,   38.280,   37.784,   37.287,   36.821,
1310     .  36.339,   35.851,   35.436,   35.037,   35.538,   35.183,   35.007,   34.953,
1311     .  34.646,   34.646,   34.854,   34.355,   34.170,   33.724,   33.538,   33.942,
1312     .  34.415,   34.716,   34.728,   34.513,   34.276,   34.241,   33.842,   34.034,
1313     .  34.004,   34.406,   34.704,   35.040,   35.438,   35.563,   35.493,   35.628,
1314     .  35.724,   35.535,   35.973,   36.356,   36.770,   37.273,   37.521,   37.141,
1315     .  36.750,   37.024,   37.249,   37.637,   37.991,   38.449,   38.888,   39.366,
1316     .  39.868,   40.371,   40.824,   40.983,   41.485,   41.434,   45.513,   45.125,
1317     .  44.710,   44.378,   44.095,   43.923,   44.188,   43.687,   43.252,   42.979/
1318        data (latitude(i),i= 9521, 9600)/
1319     .  42.969,   42.663,   42.221,   42.194,   42.461,   42.559,   42.565,   42.068,
1320     .  41.649,   42.069,   42.542,   42.869,   43.367,   43.158,   43.621,   44.076,
1321     .  44.574,   45.067,   45.506,   43.587,   43.587,   34.688,   34.686,   34.237,
1322     .  34.230,   33.830,   33.824,   33.311,   33.307,   33.077,   33.069,   34.538,
1323     .  34.530,   34.779,   34.777,   33.144,   33.141,   34.096,   34.086,   45.243,
1324     .  45.246,   45.451,   45.449,   42.211,   42.214,   38.287,   37.811,   38.300,
1325     .  38.281,   36.328,   36.318,   32.694,   32.219,   31.749,   31.249,   31.642,
1326     .  31.161,   31.384,   31.868,   32.351,   32.458,   32.349,   32.356,   32.693,
1327     .  32.697,   30.828,   30.355,   30.838,   30.823,   30.445,   30.443,   29.993,
1328     .  29.992,   29.874,   29.872,   29.653,   29.653,   28.356,   28.356,   28.500/
1329        data (latitude(i),i= 9601, 9680)/
1330     .  28.223,   28.509,   28.498,   28.191,   28.191,   27.899,   27.901,   27.415,
1331     .  27.417,   26.859,   26.456,   26.876,   26.855,   26.386,   26.386,   25.869,
1332     .  25.870,   24.866,   24.859,   24.438,   24.439,   24.411,   24.405,   24.469,
1333     .  24.470,   24.483,   24.483,   30.480,   30.480,   27.733,   27.733,   27.621,
1334     .  27.621,   27.174,   27.174,   27.079,   27.079,   26.712,   26.712,   24.789,
1335     .  24.789,   25.446,   25.446,   20.024,   20.024,   27.236,   27.236,   44.509,
1336     .  44.052,   44.507,   43.536,   43.535,   43.866,   43.861,   53.153,   52.650,
1337     .  52.154,   51.658,   51.215,   50.719,   50.217,   49.716,   49.214,   48.736,
1338     .  48.272,   47.774,   47.276,   46.776,   46.274,   46.608,   46.588,   46.119,
1339     .  46.616,   47.008,   47.459,   47.944,   48.432,   48.910,   49.304,   49.156/
1340        data (latitude(i),i= 9681, 9760)/
1341     .  48.753,   49.204,   49.690,   50.169,   50.656,   51.152,   51.641,   52.117,
1342     .  52.604,   52.803,   48.291,   48.291,   46.211,   46.243,   52.803,   53.301,
1343     .  53.768,   54.268,   53.782,   53.461,   53.153,   54.903,   54.907,   54.774,
1344     .  54.939,   54.782,   54.776,   59.018,   59.023,   59.317,   59.318,   45.514,
1345     .  45.199,   44.961,   44.550,   45.012,   45.416,   45.503,   45.514,   45.603,
1346     .  45.866,   45.603,   47.127,   47.125,   47.306,   47.303,   47.721,   47.723,
1347     .  48.049,   48.051,   48.735,   48.736,   49.168,   49.169,   49.291,   49.296,
1348     .  49.767,   49.767,   50.879,   50.884,   50.054,   50.299,   50.094,   50.052,
1349     .  50.629,   50.627,   55.327,   54.829,   55.326,   54.863,   54.510,   54.865,
1350     .  54.865,   58.559,   58.809,   58.562,   58.561,   70.799,   70.997,   71.002/
1351        data (latitude(i),i= 9761, 9840)/
1352     .  71.501,   71.556,   71.536,   71.191,   70.799,   69.768,   69.978,   69.567,
1353     .  69.763,   70.658,   70.659,   70.737,   70.746,   70.821,   70.819,   71.509,
1354     .  71.506,   74.248,   74.246,   73.892,   73.901,   73.875,   73.455,   73.290,
1355     .  73.436,   73.848,   73.878,   75.586,   75.090,   74.786,   74.855,   75.238,
1356     .  75.336,   75.589,   76.634,   76.637,   74.807,   74.965,   74.922,   75.297,
1357     .  75.027,   75.462,   75.861,   76.082,   75.622,   75.997,   75.967,   75.468,
1358     .  74.981,   74.660,   74.825,   75.374,   75.379,   72.429,   72.235,   72.503,
1359     .  72.429,   72.629,   72.632,   72.651,   72.667,   72.659,   72.653,   72.871,
1360     .  72.871,   72.401,   72.743,   73.063,   73.431,   73.056,   72.566,   72.402,
1361     .  72.971,   72.779,   72.471,   72.887,   73.385,   73.676,   73.558,   73.055/
1362        data (latitude(i),i= 9841, 9920)/
1363     .  72.973,   73.838,   73.838,   73.039,   73.041,   74.296,   74.296,   74.093,
1364     .  74.094,   76.368,   76.370,   76.605,   76.609,   77.231,   77.231,   77.387,
1365     .  77.390,   78.127,   78.127,   78.214,   78.214,   77.954,   78.190,   78.303,
1366     .  78.751,   79.149,   79.075,   78.589,   78.116,   77.955,   79.249,   79.743,
1367     .  80.098,   80.101,   79.651,   79.174,   78.929,   78.832,   79.247,   81.290,
```

150

```
1368    .  80.978,   80.495,   80.029,   80.213,   80.662,   81.132,   81.290,   81.206,
1369    .  81.209,   80.954,   80.955,   79.544,   79.544,   80.044,   79.834,   80.062,
1370    .  80.044,   77.164,   77.199,   77.202,   77.199,   76.717,   76.717,   76.263,
1371    .  76.263,   76.706,   76.706,   75.993,   75.993,   75.928,   75.928,   75.500,
1372    .  75.496,   75.138,   75.138,   74.989,   74.986,   74.810,   74.813,   74.506/
1373    data (latitude(i),i= 9921,10000)/
1374    .  74.506,   74.543,   74.541,   74.500,   74.503,   74.025,   74.026,   74.079,
1375    .  74.084,   74.159,   74.159,   74.606,   74.605,   72.893,   72.902,   72.329,
1376    .  72.344,   73.172,   73.182,   73.527,   73.528,   73.410,   73.420,   73.026,
1377    .  73.033,   73.038,   73.492,   73.015,   73.051,   69.892,   70.394,   70.095,
1378    .  69.902,   69.181,   69.196,   69.083,   69.509,   69.241,   68.815,   69.050,
1379    .  69.089,   70.977,   70.987,   71.781,   72.153,   72.645,   73.136,   73.386,
1380    .  73.238,   72.802,   72.301,   71.924,   71.925,   71.462,   71.023,   70.573,
1381    .  70.683,   70.752,   71.203,   71.576,   71.805,   74.687,   74.432,   73.960,
1382    .  73.474,   73.304,   73.584,   74.086,   74.482,   74.574,   74.574,   74.720,
1383    .  75.161,   75.483,   75.719,   75.947,   76.234,   76.670,   76.951,   76.532/
1384    data (latitude(i),i=10001,10080)/
1385    .  76.530,   76.310,   76.260,   76.060,   75.726,   75.385    74.999,   74.689,
1386    .  80.007,   80.007,   80.107,   80.111,   80.122,   80.127,   80.209,   80.219,
1387    .  80.089,   80.520,   80.075,   80.098,   80.629,   80.808,   80.592,   80.627,
1388    .  80.541,   80.887,   80.454,   80.541,   80.839,   81.166,   80.718,   80.844,
1389    .  81.004,   81.007,   81.103,   81.103,   80.827,   80.827,   81.023,   80.851,
1390    .  81.024,   80.719,   80.723,   81.216,   81.219,   81.473,   81.473,   81.730,
1391    .  81.730,   81.608,   81.608,   81.686,   81.691,   66.791,   66.790,   65.189,
1392    .  65.181,   59.043,   59.042,   59.084,   59.082,   58.675,   58.670,   58.150,
1393    .  58.143,   58.597,   58.220,   58.590,   46.244,   46.211,   46.366,   46.146,
1394    .  46.348,   46.366,   45.073,   45.073,   39.593,   39.593,   39.092,   39.092/
1395    data (latitude(i),i=10081,10160)/
1396    .  45.044,   45.044,   45.579,   45.579,   39.328,   38.942,   38.440,   37.939,
1397    .  37.532,   37.476,   37.111,   36.830,   36.660,   36.601,   36.728,   36.861,
1398    .  36.913,   37.415,   37.915,   38.419,   38.909,   39.005,   33.316,   33.311,
1399    .  13.262,   12.956,   12.579,   12.791,   13.192,   13.262,   51.508,   51.035,
1400    .  50.554,   51.045,   51.515,   55.724,   55.255,   54.760,   54.268,   53.799,
1401    .  53.326,   52.981,   52.705,   52.440,   52.111,   51.705,   51.546,   51.511,
1402    .  51.868,   52.089,   52.505,   52.817,   53.247,   53.632,   53.996,   54.462,
1403    .  54.912,   55.391,   55.746,   46.429,   46.774,   46.727,   46.798,   46.594,
1404    .  46.633,   46.473,   46.792,   46.378,   46.379,   46.458,   46.575,   46.465,
1405    .  46.328,   46.016,   45.533,   45.028,   45.438,   45.942,   46.368,   46.429/
1406    data (latitude(i),i=10161,10240)/
1407    .  44.736,   45.158,   45.481,   45.953,   46.397,   46.765,   46.679,   62.861,
1408    .  62.586,   62.089,   61.631,   61.144,   61.049,   61.530,   61.886,   62.384,
1409    .  62.094,   62.572,   62.877,   62.862,   58.952,   58.555,   58.091,   58.441,
1410    .  58.867,   58.571,   59.062,   58.938,   61.731,   61.394,   61.092,   60.627,
1411    .  60.171,   60.058,   60.432,   60.894,   61.384,   61.729,   60.592,   60.592,
1412    .  69.287,   69.286,   46.680,   46.195,   46.080,   46.220,   46.165,   45.811,
1413    .  45.379,   45.725,   46.036,   46.220,   39.328,   39.827,   40.264,   40.222,
1414    .  40.586,   40.866,   41.321,   41.738,   42.116,   42.534,   42.958,   43.456,
1415    .  43.952,   44.366,   44.747,   44.747,   45.163,   45.633,   45.767,   46.057,
1416    .  46.469,   46.614,   46.882,   47.113,   46.884,   46.949,   46.658,   46.155/
1417    data (latitude(i),i=10241,10320)/
1418    .  46.031,   46.033,   45.592,   45.380,   45.318,   44.906,   44.571,   44.634,
1419    .  44.130,   43.661,   43.169,   42.901,   42.812,   42.312,   41.814,   41.367,
1420    .  41.863,   42.109,   41.794,   41.343,   40.982,   40.680,   40.886,   40.998,
1421    .  40.524,   40.023,   40.004,   39.545,   39.253,   39.056,   39.005,   46.079,
1422    .  45.938,   45.477,   45.008,   44.508,   44.020,   43.745,   43.663,   43.603,
1423    .  43.754,   44.257,   44.678,   44.736,  -21.008,  -21.503,  -21.966,  -22.427,
1424    . -22.411,  -22.634,  -23.138,  -23.572,  -23.949,  -24.151,  -24.151,  -24.605,
1425    . -24.978,  -25.312,  -25.807,  -26.305,  -26.802,  -27.295,  -27.723,  -27.953,
1426    . -27.952,  -28.444,  -28.943,  -29.425,  -29.920,  -30.396,  -30.895,  -31.388,
1427    . -31.863,  -32.336,  -32.444,  -32.443,  -32.796,  -33.122,  -33.554,  -34.009,
1428    data (latitude(i),i=10321,10400)/
1429    . -34.490,  -34.980,  -35.406,  -35.860,  -36.087,  -36.086,  -36.591,  -37.078,
1430    . -37.565,  -37.793,  -37.814,  -37.914,  -38.209,  -38.578,  -38.695,  -39.143,
1431    . -39.143,  -38.819,  -38.464,  -38.409,  -37.933,  -38.109,  -38.517,  -38.858,
```

151

```
1432        . -38.861, -38.859, -38.629, -38.375, -38.268, -38.144, -38.008, -37.611,
1433        . -37.220, -36.956, -36.957, -36.460, -36.013, -35.659, -36.001, -35.633,
1434        . -35.653, -35.653, -35.649, -35.167, -34.665, -34.216, -34.669, -35.152,
1435        . -35.243, -34.825, -34.324, -33.838, -33.376, -32.876, -33.182, -33.648
1436        . -33.878, -34.243, -34.630, -35.013, -35.010, -34.733, -34.238, -33.791,
1437        . -33.337, -33.106, -32.606, -32.270, -32.211, -31.954, -32.029, -32.028,
1438        . -31.833, -31.574, -31.605, -31.576, -31.626, -31.680, -31.899, -32.094/
1439        data (latitude(i),i=10401,10480)/
1440        . -32.239, -32.307, -32.247, -32.274, -40.745, -41.206, -41.712, -42.215,
1441        . -42.604, -43.105, -43.231, -43.233, -42.825, -43.262, -43.614, -43.539,
1442        . -43.104, -42.673, -42.170, -41.745, -41.405, -41.405, -40.909, -40.711,
1443        . -41.011, -41.171, -40.979, -40.831, -40.744, -32.272, -32.551, -32.764,
1444        . -32.945, -33.331, -33.774, -34.003, -33.894, -33.826, -33.819, -33.885,
1445        . -33.937, -34.097, -34.389, -34.388, -34.452, -34.739, -35.015, -35.021,
1446        . -35.035, -34.850, -34.517, -34.362, -33.879, -33.522, -33.524, -33.417,
1447        . -32.919, -32.424, -31.918, -31.441, -30.987, -30.533, -30.040, -29.540,
1448        . -29.064, -28.789, -28.789, -28.321, -27.878, -27.378, -26.932, -26.533,
1449        . -26.048, -26.485, -25.984, -26.313, -25.888, -25.472, -25.037, -24.894/
1450        data (latitude(i),i=10481,10560)/
1451        . -24.894, -24.435, -23.935, -23.510, -23.009, -22.520, -22.050, -22.509,
1452        . -22.149, -21.753, -21.569, -21.260, -20.999, -20.999, -20.747, -20.624,
1453        . -20.661, -20.410, -20.311, -20.022, -20.069, -19.973, -19.973, -19.907,
1454        . -19.715, -19.465, -19.061, -18.625, -18.270, -17.767, -17.266, -16.937,
1455        . -16.521, -16.596, -16.596, -17.018, -17.464, -17.058, -16.651, -16.157,
1456        . -16.273, -16.399, -15.896, -15.399, -15.521, -15.520, -15.519, -15.156,
1457        . -15.141, -14.639, -14.232, -14.641, -14.196, -14.107, -13.727, -13.726,
1458        . -13.950, -14.318, -14.676, -15.179, -14.794, -14.848, -14.927, -14.790,
1459        . -14.526, -14.027, -13.579, -13.324, -12.923, -12.923, -12.711, -12.351,
1460        . -12.296, -12.291, -12.138, -11.671, -11.489, -11.512, -11.499, -11.499/
1461        ata (latitude(i),i=10561,10640)/
1462        . -11.792, -11.905, -12.062, -12.147, -12.111, -11.819, -12.299, -12.459,
1463        . -11.989, -11.882, -11.882, -12.269, -12.714, -13.183, -13.278, -13.780,
1464        . -14.236, -14.660, -14.724, -14.726, -15.103, -15.372, -15.724, -15.896,
1465        . -16.159, -16.454, -16.751, -16.959, -17.225, -17.512, -17.572, -17.571,
1466        . -17.688, -17.470, -17.001, -16.573, -16.103, -15.602, -15.131, -14.631,
1467        . -14.134, -13.640, -13.352, -13.352, -12.859, -12.358, -11.931, -11.452,
1468        . -10.953, -10.744, -11.216, -11.710, -11.973, -11.974, -12.474, -12.925,
1469        . -13.419, -13.920, -14.395, -14.300, -14.561, -14.842, -14.946, -14.944,
1470        . -15.434, -15.922, -16.421, -16.816, -17.259, -17.745, -18.238, -18.532,
1471        . -18.532, -19.026, -19.298, -19.415, -19.829, -19.947, -20.198, -20.699/
1472        data (latitude(i),i=10641,10720)/
1473        . -21.000, -21.007, -12.183, -12.183, -12.163, -12.163,   4.170,   4.169,
1474        .   4.380,   4.741,   4.939,   5.110,   5.494,   5.782,   5.749,   5.867,
1475        .   5.867,   6.295,   6.625,   6.586,   6.399,   5.981,   5.550,   5.354,
1476        .   4.902,   4.902,   4.818,   4.891,   4.590,   4.149,   3.790,   3.394,
1477        .   3.085,   2.963,   2.869,   2.547,   2.071,   1.632,   1.626,   1.695,
1478        .   2.038,   2.064,   4.493,   4.492,   4.675,   4.676,   6.671,   6.672,
1479        .   7.294,   7.298,   7.232,   7.241,   5.276,   5.276,   2.490,   2.497,
1480        .   4.903,   4.818,   4.891,   4.711,   4.590,   5.571,   5.582,   5.274,
1481        .   5.218,   5.221,   5.103,   4.725,   4.322,   3.937,   3.653,   3.384,
1482        .   3.074,   2.602,   2.301,   2.026,   1.821,   1.821,   1.702,   1.534/
1483        data (latitude(i),i=10721,10800)/
1484        .   1.119,   0.741,   0.609,   0.245,   0.398,   0.341,  -0.141,  -0.257,
1485        .  -0.259,  -0.743,  -1.033,  -1.038,  -1.519,  -1.995,  -2.483,  -2.349,
1486        .  -2.623,  -2.966,  -3.453,  -3.953,  -4.458,  -4.958,  -5.457,  -5.660,
1487        .  -5.660,  -5.743,  -5.487,  -5.453,  -5.051,  -4.796,  -4.507,  -4.182,
1488        .  -3.793,  -3.442,  -3.100,  -2.680,  -2.315,  -1.825,  -1.368,  -1.044,
1489        .  -1.043,  -0.631,  -0.240,   0.149,   0.496,   0.968,   1.438,   1.894,
1490        .   2.163,   2.369,   2.867,   3.231,   3.606,   3.800,   4.162,   4.512,
1491        .   4.889,   5.360,   5.569,   5.706,   5.703,   5.839,   5.838,   2.359,
1492        .   2.596,   2.385,   2.355,   2.054,   2.054,   2.057,   2.054,   1.625,
1493        .   1.623,  -0.317,  -0.321,  -0.460,  -0.466,  -1.773,  -1.494,  -1.008,
1494        data (latitude(i),i=10801,10880)/
1495        .  -1.286,  -1.737,  -1.772,  -2.374,  -2.376,  -2.757,  -2.759,  -3.067,
```

152

```
1496    .  -3.067,  -5.485,  -5.487,   0.933,   0.950,   1.369,   1.265,   0.926,
1497    .  -5.744,  -5.744,  -6.534,  -6.533,  -2.825,  -2.825,  -2.903,  -2.901,
1498    .  -1.491,  -1.904,  -2.393,  -2.560,  -3.059,  -2.903,  -2.517,  -2.122,
1499    .  -2.072,  -1.634,  -1.500,  -1.492,  -2.558,  -2.681,  -3.169,  -3.047,
1500    .  -2.558,  -2.557,  -0.270,  -0.273,  -0.388,  -0.389,  -0.347,  -0.348,
1501    .  -0.002,  -0.005,   0.212,   0.209,   0.162,   0.159,   0.366,   0.360,
1502    .   0.708,   0.706,   0.835,   0.835,   0.974,   0.969,   1.025,   1.024,
1503    .   1.142,   1.135,   0.956,   0.946,   0.791,   0.789,   0.794,   0.786,
1504    .   0.797,   0.795,   0.742,   0.740,   1.025,   1.025,   0.593,   0.589/
1505        data (latitude(i),i=10881,10960)/
1506    .   0.859,   1.023,   0.877,   0.854,   0.796,   0.773,   0.799,   0.786,
1507    .   1.217,   1.206,   1.448,   1.573,   1.451,   1.436,   1.945,   1.940,
1508    .   2.838,   2.844,   2.736,   2.743,   3.281,   3.286,   3.154,   3.155,
1509    .   4.692,   4.693,   3.850,   3.852,   2.982,   2.984,   2.898,   2.900,
1510    .   2.533,   2.540,   1.013,   1.014,  -1.031,  -1.033,  -1.570,  -1.571,
1511    .  -4.760,  -4.764,  -3.281,  -3.780,  -3.300,  -3.285,  -3.405,  -3.416,
1512    .   2.313,   2.313,   3.430,   3.433,   3.578,   3.580,   3.572,   3.574,
1513    .   4.102,   4.102,   4.258,   4.261,   2.064,   1.718,   1.292,   0.822,
1514    .   0.323,  -0.105,  -0.588,  -0.969,  -1.293,  -1.782,  -2.248,  -2.521,
1515    .  -2.521,  -2.901,  -3.051,  -3.009,  -3.446,  -3.319,  -3.281,  -3.254/
1516        data (latitude(i),i=10961,11040)/
1517    .  -3.469,  -3.334,  -3.349,  -3.349,  -3.852,  -3.979,  -3.794,  -3.546,
1518    .  -3.124,  -2.634,  -2.191,  -1.812,  -1.427,  -1.165,  -0.880,  -0.525,
1519    .  -0.524,  -0.028,   0.470,   0.784,   0.829,   0.956,   1.311,   1.598,
1520    .   1.956,   2.444,   2.804,   3.195,   3.600,   3.704,   4.169,  -7.684,
1521    .  -7.684,  -7.838,  -8.059,  -8.206,  -8.264,  -8.270,  -8.343,  -8.398,
1522    .  -8.356,  -8.554,  -8.639,  -8.140,  -7.708,  -7.719,  -7.730,  -7.347,
1523    .  -7.046,  -7.046,  -6.911,  -6.724,  -6.700,  -6.501,  -6.952,  -6.927,
1524    .  -6.848,  -6.850,  -6.550,  -6.341,  -6.206,  -5.981,  -6.019,  -6.018,
1525    .  -6.042,  -5.903,  -6.367,  -6.763,   6.829,  -6.976,  -7.422,  -7.481,
1526    .  -7.656,  -7.787,  -7.676,  -7.694,  -7.796,  -7.840,  -5.732,  -5.731/
1527        data (latitude(i),i=11041,11120)/
1528    .  -6.839,  -6.839,  -8.092,  -8.187,  -8.173,  -8.640,  -8.432,  -8.117,
1529    .  -8.090,  -8.669,  -8.669,  -8.219,  -8.638,  -8.919,  -8.434,  -8.218,
1530    .  -8.386,  -8.573,  -8.659,  -8.330,  -8.372,  -8.297,  -8.626,  -8.748,
1531    .  -8.841,  -8.923,  -9.035,  -8.741,  -8.375,  -8.387,  -8.134,  -8.136,
1532    .  -9.433,  -9.374,  -9.412,  -9.662, -10.032, -10.260,  -9.974,  -9.755,
1533    .  -9.623,  -9.429,  -8.430,  -8.430,  -8.268,  -8.328,  -8.467,  -8.489,
1534    .  -8.641,  -8.385,  -8.749,  -8.849,  -8.914,  -8.827,  -8.821,  -8.784,
1535    .  -8.366,  -8.269,  -8.289,  -8.286,  -8.273,  -8.226,  -8.567,  -8.271,
1536    .  -8.345,  -8.342,  -8.136,  -8.167,  -8.446,  -8.136, -10.547, -10.545,
1537    . -10.761, -10.509, -10.886, -10.760, -10.257, -10.257, -10.156, -10.153/
1538        data (latitude(i),i=11121,11200)/
1539    .  -9.462,  -9.226,  -9.074,  -8.911,  -8.686,  -8.459,  -8.510,  -8.561,
1540    .  -8.820,  -9.125,  -9.182,  -9.184,  -9.354,  -9.745, -10.226, -10.279,
1541    . -10.107,  -9.741,  -9.461,  -8.144,  -8.150,  -7.661,  -7.606,  -7.888,
1542    .  -7.658,  -7.512,  -7.512,  -8.105,  -8.104,  -7.065,  -7.065,  -7.798,
1543    .  -7.799,  -7.116,  -7.621,  -8.020,  -7.526,  -7.142,  -7.117,  -8.084,
1544    .  -8.085,  -7.116,  -7.115,  -5.784,  -6.286,  -5.786,  -5.787,  -5.591,
1545    .  -5.653,  -5.193,  -4.711,  -4.219,  -3.736,  -3.517,  -3.083,  -2.619,
1546    .  -2.160,  -1.676,  -1.178,  -0.771,  -0.878,  -0.877,  -0.379,   0.110,
1547    .   0.583,   0.808,   1.182,   1.247,   1.081,   1.009,   0.871,   0.815,
1548    .   0.816,   0.913,   0.829,   1.017,   1.412,   1.733,   1.238,   0.858/
1549        data (latitude(i),i=11201,11280)/
1550    .   0.452,   0.333,   0.304,   0.511,   0.511,   0.493,   0.480,   0.529,
1551    .   0.410,   0.528,   0.240,  -0.232,  -0.719,  -1.018,  -1.368,  -1.367,
1552    .  -1.191,  -0.849,  -0.931,  -0.771,  -0.601,  -0.947,  -0.905,  -1.283,
1553    .  -1.616,  -1.858,  -1.994,  -1.994,  -2.254,  -2.695,  -3.081,  -3.565,
1554    .  -3.884,  -4.284,  -4.429,  -4.827,  -4.764,  -4.268,  -4.044,  -4.045,
1555    .  -3.796,  -3.349,  -2.867,  -2.678,  -2.965,  -3.422,  -3.919,  -4.421,
1556    .  -4.918,  -5.412,  -5.592,   1.521,   1.526,   2.333,   2.329,   2.759,
1557    .   2.753,   3.429,   3.423,   3.757,   3.754,   3.903,   3.900,   4.477,
1558    .   4.006,   4.508,   4.471,  -0.146,  -0.152,  -0.472,  -0.474,  -0.377,
1559    .  -0.379,  -0.336,  -0.345,  -0.266,  -0.265,  -1.305,  -1.624,  -1.212/
```

```
1560        data (latitude(i),i=11281,11360)/
1561     .   -1.226,   -1.311,   -1.841,   -1.843,   -1.689,   -1.691,   -1.949,   -1.954,
1562     .   -1.658,   -1.700,   -2.013,   -1.657,   -1.784,   -1.791,   -1.953,   -1.784,
1563     .   -1.979,   -2.467,   -2.004,   -1.979,   -3.549,   -3.547,   -3.978,   -3.979,
1564     .   -5.078,   -5.081,   -4.618,   -5.115,   -4.616,   -4.620,   -4.404,   -4.906,
1565     .   -5.402,   -5.684,   -5.198,   -4.697,   -4.406,   -3.120,   -3.061,   -3.219,
1566     .   -3.711,   -3.767,   -3.468,   -3.123,   -2.860,   -2.855,   -2.959,   -2.834,
1567     .   -2.989,   -3.195,   -3.621,   -3.629,   -3.364,   -3.445,   -3.351,   -3.456,
1568     .   -3.446,   -2.959,   -2.859,   -3.594,   -3.593,   -3.523,   -3.520,   -3.496,
1569     .   -3.494,   -3.648,   -3.648,   -4.503,   -4.510,   -3.963,   -3.965,   -4.432,
1570     .   -4.437,   -4.694,   -4.694,   -5.677,   -5.679,   -5.301,   -5.755,   -5.293/
1571        data (latitude(i),i=11361,11440)/
1572     .   -5.304,   -6.170,   -6.512,   -6.922,   -6.427,   -6.172,   -5.434,   -5.872,
1573     .   -6.356,   -5.956,   -5.484,   -5.437,   -1.435,   -1.480,   -1.739,   -1.435,
1574     .   -1.176,   -1.178,   -0.309,   -0.711,   -0.566,   -0.310,   -0.615,   -0.614,
1575     .   -0.266,   -0.268,    0.378,    0.374,    0.740,    0.738,    0.852,    0.851,
1576     .    2.014,    2.506,    2.010,    2.017,    0.202,    0.398,    0.468,   -0.032,
1577     .   -0.498,   -0.882,   -0.611,   -0.226,   -0.214,   -0.214,    0.288,    0.774,
1578     .    1.240,    1.718,    2.118,    1.634,    1.433,    1.432,    1.042,    1.159,
1579     .    1.532,    1.066,    0.824,    0.551,    0.201,    0.033,    0.032,   -0.010,
1580     .   -0.164,   -0.244,   -0.240,   -0.053,   -0.009,   -0.418,   -0.419,   -0.770,
1581     .   -0.773,   -0.893,   -0.891,   -1.146,   -1.146,   -1.680,   -2.008,   -1.678/
1582        data (latitude(i),i=11441,11520)/
1583     .   -1.685,   -2.625,   -2.621,   -4.105,   -4.107,   -7.379,   -7.559,   -8.041,
1584     .   -8.348,   -8.380,   -7.894,   -7.503,   -7.377,   -8.168,   -8.177,   -9.128,
1585     .   -8.810,   -8.438,   -7.989,   -8.154,   -8.165,   -7.662,   -7.255,   -6.976,
1586     .   -6.974,   -6.974,   -6.743,   -6.341,   -5.841,   -5.463,   -5.213,   -4.996,
1587     .   -4.845,   -4.669,   -4.495,   -4.468,   -4.259,   -4.259,   -3.924,   -3.827,
1588     .   -3.394,   -3.799,   -4.106,   -3.685,   -3.194,   -2.922,   -2.801,   -2.471,
1589     .   -2.565,   -2.718,   -2.718,   -2.292,   -2.240,   -2.283,   -2.222,   -1.988,
1590     .   -1.527,   -1.410,   -0.910,   -0.789,   -0.789,   -0.584,   -0.351,   -0.453,
1591     .   -0.697,   -0.753,   -1.161,   -1.662,   -2.161,   -2.614,   -2.861,   -2.863,
1592     .   -3.168,   -3.386,   -3.131,   -2.708,   -2.328,   -2.204,   -1.954,   -1.954/
1593        data (latitude(i),i=11521,11600)/
1594     .   -1.591,   -1.596,   -1.761,   -1.974,   -2.181,   -2.375,   -2.408,   -2.613,
1595     .   -2.608,   -1.748,   -1.751,   -2.013,   -2.018,   -2.324,   -2.334,   -0.941,
1596     .   -0.943,   -1.480,   -1.481,   -1.597,   -1.647,   -1.722,   -1.818,   -1.688,
1597     .   -1.600,   -0.658,   -0.784,   -1.093,   -1.142,   -0.771,   -0.662,   -2.608,
1598     .   -2.761,   -2.964,   -3.161,   -3.317,   -3.403,   -3.641,   -3.809,   -4.041,
1599     .   -4.373,   -4.619,   -5.057,   -5.506,   -5.651,   -5.898,   -5.971,   -6.330,
1600     .   -6.751,   -6.740,   -6.748,   -6.747,   -7.217,   -7.602,   -7.946,   -8.290,
1601     .   -8.721,   -9.098,   -9.026,   -9.525,   -9.620,  -10.078,  -10.209,  -10.689,
1602     .  -10.544,  -10.463,  -10.463,  -10.345,  -10.238,  -10.181,  -10.138,   -9.965,
1603     .   -9.546,   -9.246,   -8.883,   -8.455,   -8.366,   -8.364,   -8.040,   -7.914/
1604        data (latitude(i),i=11601,11680)/
1605     .   -7.777,   -7.576,   -7.704,   -8.005,   -8.299,   -8.283,   -8.464,   -8.869,
1606     .   -9.145,   -9.252,   -9.192,   -9.197,   -9.129,   -8.426,   -8.696,   -8.451,
1607     .   -8.426,   -8.359,   -8.358,  -10.539,  -10.539,  -11.323,  -11.519,  -11.363,
1608     .  -11.323,  -11.353,  -11.353,  -10.623,  -10.623,  -10.598,  -10.598,   -9.712,
1609     .   -9.929,   -9.759,   -9.712,   -9.342,   -9.627,   -9.456,   -9.342,   -8.977,
1610     .   -8.977,   -8.432,   -8.432,   -9.214,   -9.214,   -5.194,   -5.194,   -5.492,
1611     .   -5.492,   -4.146,   -4.602,   -4.987,   -5.486,   -5.540,   -5.938,   -6.134,
1612     .   -6.296,   -6.290,   -6.058,   -5.918,   -5.749,   -5.533,   -5.586,   -5.527,
1613     .   -5.064,   -5.546,   -5.551,   -5.227,   -4.935,   -4.456,   -4.206,   -4.146,
1614     .   -2.564,   -2.873,   -3.062,   -3.328,   -3.651,   -3.987,   -4.450,   -4.141
1615        data (latitude(i),i=11681,11760)/
1616     .   -3.728,   -3.456,   -3.147,   -2.874,   -2.564,   -2.909,   -2.909,   -2.466,
1617     .   -2.518,   -2.478,   -2.460,   -1.353,   -1.353,   -2.596,   -2.596,   -2.712,
1618     .   -2.712,   -3.043,   -3.043,   -3.446,   -3.446,   -4.043,   -4.043,   -4.371,
1619     .   -4.371,   -3.435,   -3.435,   -4.540,   -4.540,   -5.436,   -5.683,   -6.096,
1620     .   -6.335,   -6.882,   -6.644,   -6.213,   -5.801,   -5.436,   -5.013,   -5.013,
1621     .   -4.648,   -4.648,   -4.868,   -4.868,   -2.237,   -2.237,   -1.961,   -2.192,
1622     .   -2.040,   -1.961,   -1.111,   -1.111,   -1.209,   -1.209,   -1.382,   -1.382,
1623     .   -1.694,   -1.694,   -4.535,   -4.535,   -4.048,   -4.048,   14.456,   14.504,
```

```
1624        .  18.406,   17.916,   17.418,   16.966,   16.510,   16.055,   15.860,   15.374,
1625        .  14.897,   14.398,   13.983,   14.281,   14.238,   13.761,   13.926,   13.449/
1626           data (latitude(i),i=11761,11840)/
1627        .  13.057,   12.560,   12.888,   13.214,   13.556,   13.925,   13.430,   13.641,
1628        .  13.944,   13.650,   13.892,   14.384,   14.855,   14.774,   15.264,   15.753,
1629        .  16.243,   16.047,   16.547,   17.031,   17.527,   18.017,   18.509,   18.628,
1630        .  18.393,   18.379,   18.397,   20.833,   20.830,   20.480,   20.475,   19.574,
1631        .  19.571,   19.384,   19.378,   19.147,   19.147,   18.891,   18.891,   18.983,
1632        .  18.979,   15.047,   15.039,   14.221,   14.218,   14.059,   13.563,   14.024,
1633        .  14.058,   12.580,    2.321,   11.954,   12.147,   12.524,   12.576,   12.652,
1634        .  12.651,   13.143,    2.758,   13.092,   13.137,   12.476,   12.470,   12.602,
1635        .  12.599,   12.654,   12.166,   12.659,   12.653,   13.556,   13.553,   13.520,
1636        .  13.500,   13.287,   12.803,   12.312,   12.517,   12.995,   13.406,   13.525/
1637           data (latitude(i),i=11841,11920)/
1638        .  13.847,   13.845,   12.579,   12.535,   12.413,   11.924,   11.430,   11.282,
1639        .  11.757,   12.082,   12.533,   12.578,   11.536,   11.427,   10.965,   10.486,
1640        .  10.011,   10.416,   10.919,   11.303,   11.536,   10.427,    9.925,   10.423,
1641        .  10.427,   10.021,   10.021,    9.805,    9.525,    9.130,    8.633,    8.189,
1642        .   7.721,    7.221,    6.785,    6.299,    6.786,    7.231,    6.898,    6.415,
1643        .   5.915,    6.108,    5.951,    6.182,    6.646,    7.146,    7.640,    7.833,
1644        .   7.527,    7.780,    7.381,    6.932,    7.425,    7.903,    8.134,    8.472,
1645        .   8.687,    8.340,    8.535,    8.767,    9.061,    9.493,    9.809,    6.725,
1646        .   6.722,    6.378,    6.377,    6.093,    6.092,    5.338,    5.341,    7.036,
1647        .   7.036,    9.289,    9.288,    9.238,    9.238,   10.102,    9.614,    9.731/
1648           data (latitude(i),i=11921,12000)/
1649        .  10.120,   10.095,   11.277,   11.275,   11.215,   10.712,   10.238,    9.822,
1650        .  10.325,   10.761,   11.240,   11.215,   10.877,   10.403,    9.931,    9.432,
1651        .   9.364,    9.743,   10.064,   10.560,   10.974,   10.877,   10.731,   10.731,
1652        .  11.897,   11.635,   11.467,   10.985,   10.670,   10.426,   10.926,   11.423,
1653        .  11.907,   11.894,   12.323,   12.066,   12.270,   12.321,   11.956,   11.956,
1654        .  11.510,   11.510,   10.636,   10.629,    8.319,    8.318,    8.056,    8.050,
1655        .   8.121,    8.121,   11.409,   10.907,   10.414,   10.079,    9.892,    9.422,
1656        .   9.136,    8.757,    8.491,    8.954,    9.262,    9.613,   10.001,   10.348,
1657        .  10.759,   11.254,   11.402,  -22.231,  -22.231,  -22.349,  -22.349,  -23.772,
1658        . -23.772,  -25.798,  -25.297,  -24.801,  -25.295,  -25.779,  -25.798,  -27.342/
1659           data (latitude(i),i=12001,12080)/
1660        . -27.342,  -27.717,  -27.717,  -38.773,  -38.773,  -39.499,  -39.499,  -38.561,
1661        . -38.561,  -38.423,  -38.423,  -36.075,  -36.045,  -35.641,  -35.759,  -36.075,
1662        . -35.074,  -35.074,  -33.772,  -33.772,  -32.322,  -32.322,  -40.111,  -39.609,
1663        . -40.109,  -40.111,  -40.267,  -39.801,  -40.156,  -40.267,  -40.474,  -40.474,
1664        . -40.587,  -40.587,  -43.254,  -43.254,  -43.489,  -43.489,  -40.709,  -40.709,
1665        . -40.469,  -40.469,  -40.592,  -40.592,  -42.731,  -42.731,  -26.099,  -25.642,
1666        . -26.067,  -26.099,  -25.266,  -25.266,  -24.965,  -24.965,  -20.857,  -20.857,
1667        . -15.439,  -15.439,  -15.309,  -15.309,  -15.022,  -15.022,  -14.611,  -14.611,
1668        . -11.811,  -11.358,  -11.731,  -11.831,  -11.811,  -11.928,  -11.659,  -11.174,
1669        . -11.354,  -11.252,  -11.740,  -11.928,  -11.314,  -11.314,  -11.715,  -11.715/
1670           data (latitude(i),i=12081,12160)/
1671        . -11.670,  -11.670,  -11.424,  -11.424,  -12.057,  -12.057,  -13.522,  -13.522,
1672        . -13.847,  -13.847,  -14.195,  -13.747,  -14.192,  -14.239,  -14.195,  -14.893,
1673        . -14.893,  -15.644,  -15.644,  -15.775,  -15.775,  -15.609,  -15.609,  -15.829,
1674        . -15.829,  -16.740,  -16.410,  -16.757,  -16.740,  -17.130,  -17.130,  -10.179,
1675        . -10.179,  -10.259,  -10.259,  -10.749,  -10.749,  -10.638,  -10.638,  -18.468,
1676        . -18.468,  -19.165,  -19.165,  -20.143,  -20.143,  -20.324,  -20.324,  -20.527,
1677        . -20.527,  -11.710,  -11.710,  -15.744,  -15.744,  -29.069,  -29.069,  -31.592,
1678        . -31.592,  -41.689,  -41.239,  -40.984,  -41.299,  -40.817,  -40.704,  -41.162,
1679        . -41.634,  -41.999,  -42.477,  -42.963,  -43.115,  -43.436,  -43.719,  -43.719,
1680        . -43.953,  -44.179,  -44.595,  -44.898,  -45.392,  -45.797,  -46.238,  -46.284/
1681           data (latitude(i),i=12161,12240)/
1682        . -46.598,  -46.639,  -46.492,  -46.116,  -45.761,  -45.266,  -44.820,  -44.787,
1683        . -44.787,  -44.294,  -44.024,  -43.550,  -43.719,  -43.539,  -43.859,  -43.900,
1684        . -43.425,  -43.359,  -43.406,  -43.006,  -42.584,  -42.178,  -41.755,  -41.687,
1685        . -47.221,  -47.220,  -46.793,  -46.794,  -46.722,  -47.186,  -46.688,  -46.731,
1686        . -48.020,  -48.023,  -50.564,  -50.568,  -52.460,  -52.466,  -49.659,  -49.669,
1687        . -47.680,  -47.681,  -44.315,  -44.310,  -44.014,  -44.012,  -45.630,  -45.638,
```

```
1688        . -40.911, -40.918, -39.683, -40.182, -40.590, -41.029, -41.412, -41.388,
1689        . -40.883, -40.409, -39.935, -39.711, -39.461, -39.014, -38.693, -38.194,
1690        . -38.101, -36.803, -36.939, -36.708, -37.205, -37.683, -37.876, -37.986,
1691        . -37.658, -37.647, -38.138, -38.629, -39.088, -39.134, -39.590, -39.681/
1692          data (latitude(i),i=12241,12320)/
1693        . -38.101, -37.611, -37.135, -36.671, -36.171, -35.876, -36.330, -35.966,
1694        . -35.561, -35.087, -34.654, -35.019, -35.048, -35.365, -35.844, -36.325,
1695        . -36.827, -36.802, -36.350, -36.348, -36.830, -36.829, -31.439, -31.431,
1696        . -30.588, -30.575, -30.203, -30.205, -29.284, -29.285, -34.176, -34.176,
1697        . -46.804, -46.806,   5.297,   5.297,   7.512,   7.512,   6.722,   6.722,
1698        .   6.744,   6.744,   5.781,   5.781,   6.827,   6.827,   6.987,   6.987,
1699        .   5.282,   5.282,   7.327,   7.327,   7.424,   7.424,   7.349,   7.349,
1700        .   6.702,   6.702,   7.361,   7.361,   7.372,   7.372,   8.613,   8.613,
1701        .   8.976,   8.976,   8.577,   8.577,   7.506,   7.506,   7.378,   7.378,
1702        .   6.706,   6.706,  -0.495,  -0.495,   6.008,   5.799,   6.079,   6.079/
1703          data (latitude(i),i=12321,12400)/
1704        .   3.006,   3.006,   3.096,   3.096,   3.260,   3.260,   1.706,   1.706,
1705        .   1.534,   1.534,   1.846,   1.846,   1.346,   1.346,   1.018,   1.018,
1706        .   0.448,   0.448,   0.208,   0.208,  -0.599,  -0.599,  -0.673,  -0.673,
1707        .  -1.154,  -1.154,  -1.439,  -1.439,  -1.830,  -1.830,  -1.289,  -1.289,
1708        .  -1.333,  -1.333,  -5.680,  -5.680,  -6.264,  -6.264,  -6.092,  -6.092,
1709        .  -7.175,  -7.175,  -7.472,  -7.472,  -8.036,  -8.036,  -8.466,  -8.547,
1710        .  -4.501,  -4.501,  -4.439,  -4.439,  -3.119,  -3.119,  -2.764,  -2.764,
1711        .  -9.943,  -9.943,  -9.867,  -9.867,  -2.487,  -2.487,  -2.608,  -2.608,
1712        .   4.628,   4.571,   6.048,   6.062,  -5.658,  -5.658,   0.228,   0.228,
1713        .   0.809,   0.809,   5.633,   5.633,  24.309,  24.309,   7.135,   7.135/
1714          data (latitude(i),i=12401,12480)/
1715        .   8.911,   8.911,   7.612,   7.612,   7.329,   7.329,   7.464,   7.464,
1716        .   8.748,   8.748,  11.157,  11.157,  11.640,  11.640,  19.330,  19.330,
1717        .   7.767,   7.767,   9.334,   9.334,   9.219,   9.219,  11.706,  11.706,
1718        .  11.354,  11.354,  11.397,  11.397,   9.633,   9.633,   9.559,   9.559,
1719        .  10.451,  10.451,  11.232,  11.232,  14.581,  14.581,   7.027,   7.074,
1720        .   7.041,   6.972,   7.068,   7.108,   7.141,   7.066,   9.385,   9.333,
1721        .  -7.324,  -7.324,  -6.968,  -6.968,  -6.791,  -6.791,  -7.389,  -7.389,
1722        .  -6.604,  -6.852,  -7.214,  -7.061,  -6.693,  -6.604,  -7.575,  -7.575,
1723        .  -7.941,  -7.944,  -7.852,  -7.852,  -8.192,  -8.192,  -8.164,  -8.164,
1724        .  -8.414,  -8.414,  -8.710,  -8.710,  -8.701,  -8.701,  -8.509,  -8.509/
1725          data (latitude(i),i=12481,12560)/
1726        .  -7.971,  -8.356,  -8.321,  -7.971,  -8.378,  -8.378,  -7.535,  -7.756,
1727        .  -7.986,  -8.276,  -8.216,  -7.965,  -7.598,  -7.535,  -7.571,  -7.571,
1728        .  -7.468,  -7.468,  -9.041,  -9.041,  -8.997,  -8.997,  -9.111,  -9.111,
1729        .  -8.996,  -8.996,  -9.064,  -9.064,  -9.254,  -9.421,  -9.626,  -9.821,
1730        .  -9.661,  -9.254,  -8.310,  -8.768,  -9.160,  -8.977,  -8.483,  -8.310,
1731        .  -9.351,  -9.351,  -8.373,  -8.373,  -9.712,  -9.712, -10.206, -10.409,
1732        . -10.654, -10.691, -10.331, -10.206, -11.290, -11.290, -11.472, -11.709,
1733        . -11.546, -11.472, -10.754, -10.754, -11.282, -11.282, -11.602, -11.602,
1734        .  -5.300,  -5.300,  -5.414,  -5.414,  -5.467,  -5.467,  -5.490,  -5.490,
1735        . -19.248, -19.248, -19.680, -19.680, -20.137, -20.137, -20.700, -20.396/
1736          data (latitude(i),i=12561,12640)/
1737        . -20.167, -20.630, -20.997, -21.362, -21.665, -21.928, -22.178, -22.374,
1738        . -21.939, -21.611, -21.329, -20.973, -20.700, -22.571, -22.571, -20.659,
1739        . -20.659, -20.723, -20.723, -20.953, -20.953, -21.558, -21.558, -22.331,
1740        . -22.331, -20.196, -20.196, -19.588, -19.588, -18.854, -18.854, -17.660,
1741        . -17.660, -16.818, -16.818, -16.198, -16.198, -15.888, -15.888, -15.303,
1742        . -15.303, -15.479, -15.479, -14.323, -14.323, -13.860, -13.860, -13.686,
1743        . -13.686, -13.554, -13.554, -13.434, -13.434, -13.175, -13.175, -14.999,
1744        . -15.494, -15.519, -15.036, -14.992, -15.610, -15.610, -15.756, -15.756,
1745        . -16.450, -16.013, -16.259, -16.450, -12.518, -12.518, -16.528, -16.539,
1746        . -16.494, -16.528, -17.003, -16.538, -16.403, -16.202, -16.155, -16.174
1747          data (latitude(i),i=12641,12720)
1748        . -16.525, -16.743, -17.003, -16.833, -16.833, -17.350, -17.350, -17.813,
1749        . -17.813, -18.105, -18.105, -17.688, -17.688, -18.264, -18.162, -17.667,
1750        . -17.373, -17.434, -17.854, -18.190, -18.264, -16.719, -16.719, -17.101,
1751        . -17.101, -17.284, -17.284, -18.496, -18.496, -18.374, -18.374, -18.928,
```

156

```
1752        .  -18.928,  -19.000,  -19.000,  -19.197,  -19.197,  -18.967,  -18.967,  -18.552,
1753        .  -18.552,  -17.775,  -17.775,  -17.250,  -17.250,  -17.431,  -17.431,  -17.177,
1754        .  -17.177,  -17.963,  -17.963,  -18.191,  -18.191,  -18.641,  -18.641,  -18.966,
1755        .  -18.966,  -19.129,  -19.129,  -19.159,  -19.159,  -19.844,  -19.844,  -20.671,
1756        .  -20.671,  -16.999,  -16.786,  -16.968,  -16.999,  -21.704,  -21.704,  -17.987,
1757        .  -17.987,  -24.328,  -24.327,  -24.664,  -24.663,    7.714,    7.714,    9.576/
1758           data (latitude(i),i=12721,12800)/
1759        .    9.576,   13.635,   13.635,   14.178,   14.178,   15.069,   15.069,   15.249,
1760        .   15.249,   16.379,   16.379,   16.706,   16.706,   17.313,   17.313,   17.606,
1761        .   17.606,   18.173,   18.173,   18.802,   18.802,   19.655,   19.655,   20.520,
1762        .   20.520,    5.315,    5.315,    6.936,    6.936,    7.050,    7.050,    7.313,
1763        .    7.313,  -13.343,  -13.343,  -13.805,  -13.805,  -14.053,  -14.053,  -14.308,
1764        .  -14.308,  -21.268,  -21.268,  -19.101,  -19.101,   -9.210,   -9.210,  -14.369,
1765        .  -14.369,  -14.273,  -14.273,  -15.624,  -15.624,  -18.667,  -18.667,  -18.667,
1766        .  -18.667,  -19.786,  -19.786,  -19.735,  -19.735,  -21.446,  -21.446,   -8.985,
1767        .   -8.985,  -21.230,  -21.230,  -21.912,  -21.912,   -8.582,   -8.582,   -9.372,
1768        .   -9.372,  -14.288,  -14.288,  -14.185,  -14.185,  -18.826,  -18.826,  -18.705/
1769           data (latitude(i),i=12801,12880)/
1770        .  -18.705,  -19.652,  -19.652,  -10.890,  -10.890,  -11.548,  -11.548,  -10.037,
1771        .  -10.037,  -10.430,  -10.430,  -10.386,  -10.386,  -18.081,  -18.081,  -18.880,
1772        .  -18.880,  -19.272,  -19.272,  -19.836,  -19.836,  -20.019,  -20.019,  -19.845,
1773        .  -19.845,  -20.184,  -20.184,  -16.674,  -16.674,  -16.904,  -16.904,  -27.556,
1774        .  -27.556,  -25.076,  -25.076,  -22.655,  -22.655,  -11.464,  -11.464,  -27.914,
1775        .  -27.914,  -10.442,  -10.442,   -9.899,   -9.899,   -9.723,   -9.723,   -9.329,
1776        .   -9.329,   -8.885,   -8.885,   -8.806,   -8.806,   -7.973,   -7.973,   -9.896,
1777        .   -9.896,   -9.930,   -9.930,  -10.094,  -10.094,   -5.618,   -5.618,   -4.055,
1778        .   -4.055,   -0.385,   -0.385,    1.717,    1.717,    3.799,    3.799,    4.706,
1779        .    4.706,  -23.139,  -23.139,  -21.709,  -21.709,  -15.749,  -15.749,  -16.165/
1780           data (latitude(i),i=12881,12960)/
1781        .  -16.165,  -17.735,  -17.735,  -17.572,  -17.572,  -16.826,  -16.826,  -16.770,
1782        .  -16.770,  -16.531,  -16.531,  -15.844,  -15.844,  -16.973,  -16.973,  -16.977,
1783        .  -16.977,  -17.661,  -17.661,  -22.514,  -22.514,  -23.401,  -23.401,  -23.873,
1784        .  -23.873,  -21.534,  -21.487,  -21.568,  -21.531,  -22.029,  -22.019,  -21.466,
1785        .  -21.466,  -21.358,  -21.342,  -18.571,  -18.485,  -18.362,  -18.269,  -17.318,
1786        .  -17.309,  -17.318,  -17.358,  -18.774,  -18.754,  -20.880,  -20.788,  -21.873,
1787        .  -21.821,  -21.871,  -21.876,  -19.656,  -19.612,  -19.164,  -19.138,  -19.219,
1788        .  -19.202,  -18.419,  -18.340,  -18.220,  -18.066,  -17.810,  -17.689,  -17.703,
1789        .  -17.786,  -15.977,  -15.981,  -15.799,  -15.831,  -16.018,  -16.110,  -16.145,
1790        .  -16.186,  -16.196,  -16.231,  -17.622,  -17.537,  -16.994,  -16.974,  -17.457/
1791           data (latitude(i),i=12961,13040)/
1792        .  -17.434,  -16.672,  -16.627,  -16.638,  -16.581,  -16.545,  -16.455,  -16.743,
1793        .  -16.712,  -16.673,  -16.645,  -16.483,  -16.417,  -16.347,  -16.325,  -16.074,
1794        .  -16.085,  -16.149,  -16.194,  -15.950,  -15.891,  -15.908,  -15.758,  -15.480,
1795        .  -15.442,  -14.512,  -14.446,  -14.624,  -14.603,  -14.621,  -14.666,  -14.347,
1796        .  -14.367,  -14.383,  -14.423,  -14.438,  -14.458,  -14.461,  -14.430,  -14.500,
1797        .  -14.436,  -15.318,  -15.378,  -15.481,  -15.536,  -15.838,  -15.883,  -16.064,
1798        .  -16.054,  -16.086,  -16.350,  -16.408,  -16.460,  -16.727,  -16.755,  -17.491,
1799        .  -17.419,  -17.406,  -17.353,  -17.386,  -17.326,  -19.931,  -19.897,  -15.825,
1800        .  -15.776,  -15.270,  -15.218,  -15.304,  -15.291,  -15.265,  -15.246,  -15.233,
1801        .  -15.206,  -15.191,  -15.142,  -15.021,  -14.988,  -15.002,  -14.922,  -14.944/
1802           data (latitude(i),i=13041,13120)/
1803        .  -14.978,  -14.900,  -14.856,  -15.043,  -15.069,  -16.482,  -16.441,  -16.453,
1804        .  -16.419,  -16.420,  -16.461,  -16.829,  -16.774,  -15.798,  -15.765,  -15.813,
1805        .  -15.792,  -21.343,  -21.324,  -21.317,  -21.301,  -19.282,  -19.277,  -19.355,
1806        .  -19.331,  -20.800,  -20.799,  -17.382,  -17.349,  -16.840,  -16.831,  -14.183,
1807        .  -14.165,  -14.185,  -14.157,  -18.131,  -18.115,  -17.970,  -17.952,  -17.612,
1808        .  -17.625,  -17.836,  -17.815,  -14.427,  -14.438,  -15.266,  -15.287,  -14.908,
1809        .  -14.899,  -15.057,  -15.084,  -14.910,  -14.936,  -14.902,  -14.873,  -15.202,
1810        .  -15.226,  -19.874,  -19.857,  -20.433,  -20.388,  -54.752,  -54.752,   28.155,
1811        .   27.833,   27.678,   27.326,   27.073,   26.771,   26.790,   26.506,   26.722,
1812        .   26.995,   27.171,   27.118,   27.139,   26.949,   26.697,   26.976,   26.950/
1813
1814           end
```

```
0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c------------------------------------------------------------------- ----------
0010          subroutine MapIt
0011    c---------- ------------------------------------------------------------------
0012    c     Draw an Earth map and overlay a flight path onto it
0013    c
0014    c     developer:    David F. Smith
0015    c     date:             February 1991
0016    c
0017    c.....common block definition files
0018
0019          include 'CrvDat.inc'
0020          include 'FntCom.inc'
0021          include 'LatCom.inc'
0022          include 'LngCom.inc'
0023          include 'MapMenu.inc'
0024          include 'OptFlg.inc'
0025          include 'PenCom.inc'
0026          include 'MapCom.inc'
0027          include 'PntAbs.inc'
0028          include 'TicDat.inc'
0029          include 'VuWind.inc'
0030          include 'WinLim.inc'
0031
0032    c.....graphics window records
0033
0034          common / MapWindow /        MapWPtr
0035          record / WindowPtr /        MapWptr
0036
0037    c.....set up pointer for QuickDraw globals
0038
0039          common  / QDGPtr    /        QDG
0040          pointer / QDGlobals /        QDG
0041
0042    c.....Picture record handle and pointer
0043
0044          common / pict       /        PictHndl
0045          record / PicHandle /        PictHndl
0046
0047    c.....pointer to off screen bit map
0048
0049          record / GrafPtr   /        OffScreen
0050
0051    c.....Handle to off screen bit map memory contents
0052
0053          record / Handle    /        myBitH
0054
0055    c.....event record
0056
0057          record / EventRecord /      TheEvent
0058
0059    c.....cursor handle
0060
0061          record / CursHandle         CursorHndl
0062
0063    c.....user option code
0064
```

158

```
0065          integer*2                  Option
0066
0067   c.....continue reading data flag
0068
0069          integer*2                  goflag
0070
0071   c.....define logical*1 to represent Boolean argument from Event monitor
0072
0073          logical*1                  AnEvent
0074
0075   c.....define an initialization flag
0076
0077          logical*1                  FirstTime
0078
0079   c------------------------------------------------------------------------
0080
0081   c.....set default graph window limits
0082
0083          iGxMin = QDG^.screenBits.bounds.left
0084          iGxMax = QDG^.screenBits.bounds.right
0085          iGyMi: = QDG^.screenBits.bounds.top
0086          iGy··· = QDG^.screenBits.bounds.bottom - 38
0087
0088   c.....get screen resolution
0089
0090          call ScreenRes ( %ref(iHRes) , %ref(iVRes) )
0091          MapHRes = float ( iHRes )
0092          MapVRes = float ( iVRes )
0093
0094   c.....get default graph window size
0095
0096          DefWidth  = float ( iGxMax ~ iGxMin )/MapHRes
0097          DefHeight = float ( iGyMax ~ iGyMin )/MapVRes
0098
0099   c.....initialize window size to the default
0100
0101          MapWidth  = DefWidth
0102          MapHeight = DefHeight
0103
0104   c.....set up the 'Map' menu
0105
0106   c       call SetUpMapMenu
0107   c       call UnloadSeg ( %loc(SetUpMapMenu) )
0108
0109   c------------------------------------------------------------------------
0110   c.....the main execution loop begins here
0111   c------------------------------------------------------------------------
0112
0113          Option = oNew
0114          FirstTime = .true.
0115
0116          do while ( Option.eq.oNew .or. Option.eq.oRedraw )
0117
0118   c.........set dialog font to Chicago ( system )
0119
0120             FntNam = 'Chicago'
0121             call GetFNum ( *val(FntNam) , FntNum )
0122             call setDAfont ( *val(FntNum) )
0123
0124   c.........show 'Map' options menu
0125
0126   c          call InsertMenu ( *val(MapMenuHndl) , *val(0) )
0127   c          call DrawMenuBar
0128
```

```
0129      c........initialize user option to Cycle (after the first pass)
0130
0131            if(FirstTime) then
0132              FirstTime = .false.
0133            else
0134              Option = oCycle
0135            end if
0136
0137      c........monitor and respond to events
0138
0139            do while ( Option.eq.oCycle )
0140              AnEvent = GetNextEvent ( %val(EveryEvent) , %ref(TheEvent) )
0141              if ( AnEvent ) then
0142                  call EventHandler ( TheEvent , Option )
0143              end if
0144            end do
0145
0146      c........clear and redraw the menu bar
0147
0148      c          call ClearMenuBar
0149      c          call DrawMenuBar
0150
0151      c........set up the map appearance via a dialog window
0152
0153            if ( Option.eq.oNew ) then
0154              call SetUpTheMap ( Option )
0155            end if
0156
0157      c........draw the map via QuickDraw
0158
0159            if ( Option.eq.oNew ) then
0160                call DrawTheMap  ( Option )
0161                call OpenBitMap  ( offScreen , myBitH , MapWPtr.WP^.portRect )
0162                call SetPort     ( %val(offScreen) )
0163                call CopyBits    ( %ref(MapWPtr.WP^.portBits) ,
0164          &                        %ref(OffScreen.GrafP^.portBits) ,
0165          &                        %ref(MapWPtr.WP^.portRect) ,
0166          &                        %ref(OffScreen.GrafP^.portRect) ,
0167          &                        %val(srcCopy) , %val(nil) )
0168            else if ( Option.eq.oReDraw ) then
0169                call SetPort     ( %val(MapWPtr) )
0170                call BeginUpdate ( %val(MapWPtr) )
0171                call CopyBits    ( %ref(OffScreen.GrafP^.portBits) ,
0172          &                        %ref(MapWPtr.WP^.portBits) ,
0173          &                        %ref(OffScreen.GrafP^.portRect) ,
0174          &                        %ref(MapWPtr.WP^.portRect) ,
0175          &                        %val(srcCopy) , %val(nil) )
0176                call EndUpdate   ( %val(MapWPtr) )
0177                call SelectWindow( %val(MapWPtr) )
0178            end if
0179
0180      c........enable the save and redraw menu items after first Map is complete
0181
0182            if ( iMadeFirstMap.eq.0 ) then
0183                call MapMenuSet ( itemSaveMap , enableTheItem )
0184                call MapMenuSet ( itemRedraw  , enableTheItem )
0185                iMadeFirstMap = 1
0186            end if
0187
0188          end do
0189
0190      c.....eliminate the Map window
0191
0192            call HideWindow ( %val(MapWptr) )
```

160

```
0193            call DisposeWindow ( %val(MapWptr) )
0194
0195    c.....get rid of 'Map' menu
0196
0197    c       call DeleteMenu ( %val(MapMenuID) )
0198    c       call DisposeMenu ( %val(MapMenuHndl ) )
0199    c       call ClearMenuBar
0200    c       call DrawMenuBar
0201
0202    c.....return to calling routine
0203
0204            return
0205            end
```

```
0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c-----------------------------------------------------------------------
0010            subroutine MapMenuSet ( item , enable )
0011    c-----------------------------------------------------------------------
0012    c       This routine lets other routines enable or disable menu items without
0013    c       the calling routines having to know about menu structures.
0014
0015            logical*2                    enable
0016            integer*2                    item
0017
0018    c.....Options menu file
0019
0020            include 'MapMenu.inc'
0021
0022    c.....either enable or disable designated item
0023
0024            if ( enable ) then
0025               call EnableItem ( %val(MapMenuHndl) , %val(item) )
0026            else
0027               call DisableItem ( %val(MapMenuHndl) , %val(item) )
0028            end if
0029
0030    c.....redraw the menu bar
0031
0032            call DrawMenuBar
0033
0034            return
0035            end
```

```
0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Main segment
0010
0011    !!S Main
```

```
0012
0013    c----------------------------------------------------------------------------
Segment Main
0014          subroutine MenuSet ( menuID, menuItem, enable )
0015    c----------------------------------------------------------------------------
0016    c     This routine lets other routines enable or disable menu items without
0017    c     the calling routines having to know about menu structures.
0018
0019    !!SETC USINGINCLUDES = FALSE
0020
0021          implicit none
0022
0023    c.....declare Boolean flag
0024
0025          logical*1 enable
0026
0027    c.....accept the input arguments
0028
0029          integer*2 menuID, menuItem
0030
0031    c.....declare a structure for getting the menu handle
0032
0033          record / MenuHandle / menu
0034
0035    c----------------------------------------------------------------------------
0036
0037    c.....get the menu's handle
0038
0039          menu.menuH = GetMHandle ( %val( menuID ))
0040
0041          if (enable) then
0042             call EnableItem  ( %val( menu ), %val( menuItem ))
0043          else
0044             call DisableItem ( %val( menu ), %val( menuItem ))
0045          endif
0046
0047    c.....display the results
0048
0049          call DrawMenuBar
0050
0051          return
0052          end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c----------------------------------------------------------------------------
0010          subroutine MovAbs ( ix , iy )
0011    c----------------------------------------------------------------------------
0012    c     Move to absolute graphics window position .
0013
0014          include 'PntAbs.inc'
0015          include 'WinLim.inc'
0016
0017          ixabs  = ix
0018          iyabs  = iy
0019
0020          call MoveTo ( %val(ixabs) , %val(iGyMax-iyabs) )
```

162

```
0021
0022        return
0023        end


0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c--------------------------------------------------------------------------
0010          subroutine MovRel ( ix , iy )
0011    c--------------------------------------------------------------------------
0012    c     Move to relative graphics window position .
0013
0014          include 'PntAbs.inc'
0015          include 'WinLim.inc'
0016
0017          ixabs  = ixabs + ix
0018          iyabs  = iyabs + iy
0019
0020          call MoveTo ( %val(ixabs) , %val(iGyMax-iyabs) )
0021
0022          return
0023          end


0001          SUBROUTINE NOGAPS( Z, X, Y, TBLV1, TBLV2, FIRST )
0002    C*******************************************************************************C
0003    C                                                                              C
0004    C     THIS PROGRAM DOES A TABLE LOOKUP FOR EAST AND NORTH WIND VECTORS          C
0005    C     AT THE ALTITUDE, LATITUDE,AND LONGITUDE SPECIFIED.                        C
0006    C                                                                              C
0007    C     THE OUTPUTS OF THIS PROGRAM ARE EAST AND NORTH WIND VECTORS.              C
0008    C                                                                              C
0009    C*******************************************************************************C
0010    C
0011          LOGICAL FIRST
0012          REAL F,V
0013          INTEGER I,IP1,NL,NU,NGRD,MGRD,MAT
0014          DIMENSION  F   ( 3000 ),  V   ( 3 ),  DDV ( 3 )
0015          DIMENSION  I   (  3 ),  IP1 (  3 ),  NL  (  3 ),  NU  ( 3 )
0016    C
0017          SAVE
0018    C
0019    C*******************************************************************************C
0020    C
0021          V(1) = Z
0022          V(2) = Y
0023          V(3) = X
0024    C
0025          I(1) = 1
0026          I(2) = 1
0027          I(3) = 1
0028    C
0029    C     READ IN THE EAST VECTOR FILE
0030    C
0031          IF(FIRST)THEN
0032          OPEN(UNIT=15,STATUS='OLD',FILE='EAST.DAT')
0033    C
```

163

```
0034          J=1
0035          READ(15,1)F(J),F(J+1),F(J+2)
0036     1    FORMAT(F8.3,F8.3,F8.3)
0037          READ(15,*)
0038          MGRD=INT(F(2))
0039          NGRD=INT(F(3))
0040          MAT=MGRD*NGRD
0041     C
0042          DO J=4,12,4
0043          READ(15,6)F(J),F(J+1),F(J+2),F(J+3)
0044          END DO
0045          READ(15,*)
0046          DO J=13,MGRD+12,4
0047          READ(15,6)F(J),F(J+1),F(J+2),F(J+3)
0048          END DO
0049          READ(15,*)
0050          DO J=MGRD+13,MGRD+NGRD+12,4
0051          READ(15,6)F(J),F(J+1),F(J+2),F(J+3)
0052          END DO
0053          READ(15,*)
0054          DO J=MGRD+NGRD+13,MGRD+NGRD+MAT+12,4
0055          READ(15,5)F(J),F(J+1),F(J+2),F(J+3)
0056          END DO
0057          DO JJ=1,8
0058          DO J=(JJ-1)*MAT+(MGRD+NGRD+MAT+13),JJ*MAT+(MGRD+NGRD+MAT+12),4
0059          READ(15,5)F(J),F(J+1),F(J+2),F(J+3)
0060          END DO
0061          END DO
0062     C
0063     5    FORMAT(F8.3,F8.3,F8.3,F8.3)
0064     6    FORMAT(F9.3,F9.3,F9.3,F9.3)
0065
0066          CLOSE(15)
0067            ENDIF
0068     C
0069     C.....BEGIN TABLE LOOKUP FOR EAST VECTOR
0070     C
0071          N1 = INT(F(1))
0072          N2 = INT(F(2))
0073          N3 = INT(F(3))
0074     C
0075     C.....COMPUTE UPPER BOUNDS ON INDICES FOR Z, Y, AND X
0076     C
0077          NU(1) = 3 + N1
0078          NU(2) = NU(1) + N2
0079          NU(3) = NU(2) + N3
0080     C
0081     C.....COMPUTE LOWER BOUNDS ON INDICES FOR Z, Y, AND X
0082     C
0083          NL(1) = 4
0084          NL(2) = NL(1) + N1
0085          NL(3) = NL(2) + N2
0086     C
0087     C.....LOOP FOR ALL THREE INDICES
0088     C
0089          DO 200 J=1,3
0090     C
0091     C.....GET INDICES IN BOUNDS
0092     C
0093          IF( I(J) .LT. NL(J)   ) I(J) = NL(J)
0094          IF( I(J) .GT. NU(J)-1 ) I(J) = NU(J) - 1
0095     C
0096     C.....FIND GREATEST LOWER BOUND ON INDEX
0097     C
```

```
0098   110   CONTINUE
0099         IP1(J) = I(J) + 1
0100         IF( V(J)   .LE. F(IP1(J)) ) GO TO 130
0101         IF( IP1(J) .EQ.    NU(J) ) GO TO 150
0102         I(J) = IP1(J)
0103         GO TO 110
0104   120   CONTINUE
0105         IF( I(J) .EQ. NL(J) ) GO TO 140
0106         I(J) = I(J) - 1
0107   130   CONTINUE
0108         IF( V(J) .LT. F(I(J)) ) GO TO 120
0109   140   CONTINUE
0110         IP1(J) = I(J) + 1
0111   150   CONTINUE
0112   C
0113   C....... GET PARTIALS OF INDEPENDENT VARIABLES
0114   C
0115         DDV(J) = ( V(J) - F(I(J)) )/( F(IP1(J)) - F(I(J)) )
0116   C
0117   200   CONTINUE
0118   C
0119   C....... GET ALL INDICES INTO COEFFICIENT ARRAY
0120   C
0121         N2 = N1*N2
0122   C
0123         NN = N1*( I(2) - NL(2) ) + N2*( I(3) - NL(3) )
0124   C
0125         N111 = NU(3) + 1 + I(1) - NL(1) + NN
0126         N211 = N111 +  1
0127         N121 = N111 + N1
0128         N221 = N121 +  1
0129         N112 = N111 + N2
0130         N212 = N112 +  1
0131         N122 = N121 + N2
0132         N222 = N122 +  1
0133   C
0134         P1A1 = DDV(1)*( F(N211) - F(N111) ) + F(N111)
0135         P1B1 = DDV(1)*( F(N221) - F(N121) ) + F(N121)
0136         P2A1 = DDV(1)*( F(N212) - F(N112) ) + F(N112)
0137         P2B1 = DDV(1)*( F(N222) - F(N122) ) + F(N122)
0138   C
0139         T11  = DDV(2)*( P1B1 - P1A1 ) + P1A1
0140         T21  = DDV(2)*( P2B1 - P2A1 ) + P2A1
0141   C
0142   C.....TBLV1 IS THE EAST VECTOR OUTPUT
0143   C
0144         TBLV1 = DDV(3)*( T21  - T11  ) + T11
0145   C
0146   C.....READ IN THE NORTH VECTOR
0147   C
0148         IF(FIRST)THEN
0149         OPEN(UNIT=15,STATUS='OLD',FILE='NORTH.DAT')
0150         J=1
0151         READ(15,101)F(J),F(J+1),F(J+2)
0152   101   FORMAT(F8.3,F8.3,F8.3)
0153         READ(15,*)
0154         MGRD=INT(F(2))
0155         NGRD=INT(F(3))
0156         MAT=MGRD*NGRD
0157   C
0158         DO J=4,12,4
0159         READ(15,106)F(J),F(J+1),F(J+2),F(J+3)
0160         END DO
0161         READ(15,*)
```

```
0162          DO J=13,MGRD+12,4
0163          READ(15,106)F(J),F(J+1),F(J+2),F(J+3)
0164          END DO
0165          READ(15,*)
0166          DO J=MGRD+13,MGRD+NGRD+12,4
0167          READ(15,106)F(J),F(J+1),F(J+2),F(J+3)
0168          END DO
0169          READ(15,*)
0170          DO J=MGRD+NGRD+13,MGRD+NGRD+MAT+12,4
0171          READ(15,105)F(J),F(J+1),F(J+2),F(J+3)
0172          END DO
0173          DO JJ=1,8
0174          DO J=(JJ-1)*MAT+(MGRD+NGRD+MAT+13),JJ*MAT+(MGRD+NGRD+MAT+12),4
0175          READ(15,105)F(J),F(J+1),F(J+2),F(J+3)
0176          END DO
0177          END DO
0178    C
0179    105   FORMAT(F8.3,F8.3,F8.3,F8.3)
0180    106   FORMAT(F9.3,F9.3,F9.3,F9.3)
0181
0182          FIRST = .FALSE.
0183          CLOSE(15)
0184             ENDIF
0195    C
0186    C.....BEGIN TABLE LOOKUP FOR NORTH VECTOR
0187    C
0188          P1A1 = DDV(1)*( F(N211) - F(N111) ) + F(N111)
0189          P1B1 = DDV(1)*( F(N221) - F(N121) ) + F(N121)
0190          P2A1 = DDV(1)*( F(N212) - F(N112) ) + F(N112)
0191          P2B1 = DDV(1)*( F(N222) - F(N122) ) + F(N122)
0192    C
0193          T11  = DDV(2)*( P1B1 - P1A1 ) + P1A1
0194          T21  = DDV(2)*( P2B1 - P2A1 ) + P2A1
0195    C
0196    C.....TBLV2 IS THE NORTH VECTOR OUTPUT
0197    C
0198          TBLV2 = DDV(3)*( T21  - T11  ) + T11
0199    C
0200          IZ = I(1)
0201          IY = I(2)
0202          IX = I(3)
0203    C
0204          RETURN
0205          END


0001    c-------------------------------------------------------------------------------
0002          integer*2 function nquant ( value , quantm )
0003    c-------------------------------------------------------------------------------
0004    c     quantize the input value to the nearest number of counts in either the
0005    c     positive or negative direction as indicated by the sign of the
0006    c     quantization factor
0007
0008    c.....halt if quantization factor is zero
0009
0010          if ( quantm.eq.0.0 ) then
0011             pause 'quantization factor cannot be zero'
0012             call exit
0013          end if
0014
0015    c.....quantize to the nearest quantum in the positive direction
0016
0017          if ( quantm.gt.0.0 ) then
0018             nquant = inint ( value/quantm + 0.5 )
```

```
0019            remain = quantm*float(nquant) - value
0020            if ( remain.eq.quantm ) then
0021               nquant = nquant - 1
0022            end if
0023         end if
0024
0025   c.....quantize to the nearest quantum in the negative direction
0026
0027         if ( quantm.lt.0.0 ) then
0028            quantp = - quantm
0029            nquant = inint ( value/quantp - 0.5 )
0030            remain = value - quantp*float(nquant)
0031            if ( remain.eq.quantp ) then
0032               nquant = nquant + 1
0033            end if
0034         end if
0035
0036         return
0037         end




0001   c------------------------------------------- --- ----------------------------------------
0002         integer*2 function ntrvl ( x , xt , nx , idir )
0003   c--------------------------------------------------- ----------------------
0004   c     determine the index in a monotonic data table associated with the input
0005   c     value.  Uses binary search algorithm.
0006
0007         integer*2              nx
0008         integer*2              idir
0009         integer*2              ixmid
0010         integer*2              ixmin
0011         integer*2              ixmax
0012         real*4                 xt(nx)
0013
0014   c.....the data table has ascerding values
0015
0016         if ( idir.gt.0 ) then
0017            if ( x.ge.xt(1) .and. x.le.xt(nx) ) then
0018               ixmin  = 1
0019               ixmax  = nx
0020               do while ( ixmax.ne.ixmin+1 )
0021                  ixmid  = ( ixmin + ixmax )/2
0022                  if ( x.ge.xt(ixmid) ) then
0023                     ixmin  = ixmid
0024                  else
0025                     ixmax  = ixmid
0026                  end if
0027               end do
0028               ntrvl  = ixmin
0029               return
0030            else if ( x.lt.xt(1) ) then
0031               ntrvl  = 0
0032               return
0033            else if     .gt.xt(nx) ) then
0034               ntrvl  = nx
0035               return
0036            end if
0037
0038   c.....the data has descending values
0039
0040         else if ( idir.lt.0 ) then
0041            if ( x.le.xt(1) .and. x.ge.xt(nx) ) then
0042               ixmin  = 1
0043               ixmax  = nx
```

```
0044                  do while ( ixmax.ne.ixmin+1 )
0045                     ixmid  = ( ixmin + ixmax )/2
0046                     if ( x.le.xt(ixmid) ) then
0047                        ixmin  = ixmid
0048                     else
0049                        ixmax  = ixmid
0050                     end if
0051                  end do
0052                  ntrvl  = ixmin
0053                  return
0054               else if ( x.gt.xt(1) ) then
0055                  ntrvl  = 0
0056                  return
0057               else if ( x.lt.xt(nx) ) then
0058                  ntrvl  = nx
0059                  return
0060               end if
0061
0062        c.....the data is not monotonic
0063
0064            else if ( idir.eq.0 ) then
0065            end if
0066
0067            return
0068            end



0001        c-----------------------------------------------------------------------
0002            integer function NumChr ( ChrStr , lngth )
0003        c-----------------------------------------------------------------------
0004        c    left justify the Character string [ChrStr] and determine the number of non
0005        c    blank characters in it.
0006
0007            character*(*) ChrStr
0008
0009            ileft  = 1
0010            iright = 1
0011
0012        c.....get leftmost non blank character
0013
0014            do i = 1 , lngth
0015               if ( ChrStr(i:i).ne.' ' ) then
0016                  ileft  = i
0017                  leave
0018               end if
0019            end do
0020
0021        c.....get rightmost non blank character
0022
0023            do i = lngth , 1 , -1
0024               if ( ChrStr(i:i).ne.' ' ) then
0025                  iright = i
0026                  leave
0027               else
0028                  if ( i.eq.1 ) then
0029                     NumChr = 0
0030                     return
0031                  end if
0032               end if
0033            end do
0034
0035        c.....get number of non blank characters
0036
0037            NumChr = iright - ileft + 1
```

168

```
0038
0039    c.....left justify the string
0040
0041          ChrStr(1:NumChr)           = ChrStr(ileft:iright)
0042          ChrStr(NumChr+1:lngth) = ' '
0043
0044          return
0045          end




0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c-----------------------------------------------------------------------
0010          subroutine OpenBitMap ( newOffScreen , myBitH , inBounds )
0011    c-----------------------------------------------------------------------
0012    c     open an off screen bit map to save the contents of the graphics window
0013
0014          record / GrafPtr /        savePort
0015          record / GrafPtr /        newPort
0016          record / GrafPtr /        newOffScreen
0017          record / GrafPort /       myNewPort
0018          record / Handle /         myBitH
0019          record / Rect /           inBounds
0020          integer*4                 mySize
0021
0022    c.....get a pointer to the current port
0023
0024          call GetPort ( *ref(savePort) )
0025
0026    c.....open a new port
0027
0028          mySize = jSizeOf ( myNewPort )
0029          newPort = NewPtr ( *val(mySize) )
0030          call OpenPort ( *val(newPort) )
0031
0032    c.....set port attributes and allocate a locked memory block
0033
0034          newPort.GrafP^.portRect          = inBounds
0035          call RectRgn ( *val(newPort.GrafP^.clipRgn) , *val(inBounds) )
0036          call RectRgn ( *val(newPort.GrafP^.visRgn)  , *val(inBounds) )
0037          newPort.GrafP^.portBits.bounds   = inBounds
0038          newPort.GrafP^.portBits.rowBytes = ( inBounds.right - inBounds.left + 15 )/16*2
0039          mySize                           = newPort.GrafP^.portBits.rowBytes*(
inBounds.bottom - inBounds.top )
0040          myBitH                           = NewHandle ( *val(mySize) )
0041          call HLock ( *val(myBitH) )
0042          newPort.GrafP^.portBits.baseAddr = myBitH.bhdl^.bptr
0043
0044    c.....erase the port since it is just memory
0045
0046          call ErasePort ( *val(inBounds) )
0047
0048    c.....save a pointer to the new off screen bit map and restore the previous
0049    c     window
0050
0051          newOffScreen = newPort
0052          call SetPort ( *val(savePort) )
0053
```

```
0054        return
0055        end



0001   !!s PenDat
0002   c------------------------------------------------------------------
0003        block data PenDat
0004   c------------------------------------------------------------------
0005   c     array of pen commands
0006   c        0 Ÿ move Ÿ pen up
0007   c        1 Ÿ draw Ÿ pen down
0008   c        2 Ÿ end of data
0009
0010        include 'PenCom.inc'
0011
0012        data (PenCommand(i),i=    1,    80)/
0013      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0014      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0015      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0016      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0017      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0018      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0019      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0020      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0021      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0022      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      /
0023        data (PenCommand(i),i=   81,   160)/
0024      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0025      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0026      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0027      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0028      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0029      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0030      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0031      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0032      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0033      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      /
0034        data (PenCommand(i),i=  161,   240)/
0035      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0036      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0037      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0038      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0039      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0040      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0041      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0042      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0043      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0044      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      /
0045        data (PenCommand(i),i=  241,   320)/
0046      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0047      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0048      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0049      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 0      ,
0050      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0051      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0052      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0053      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0054      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0055      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      /
0056        data (PenCommand(i),i=  321,   400)/
0057      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0058      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0059      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0060      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
```

```
0061        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0062        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0063        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0064        . 0      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0065        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0066        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      /
0067        data (PenCommand(i),i=  401,  480)/
0068        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0069        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0070        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0071        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0072        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0073        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0074        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0075        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0076        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0077        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0078        data (PenCommand(i),i=  481,  560)/
0079        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0080        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0081        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0082        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0083        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0084        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0085        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0086        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0087        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0088        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      /
0089        data (PenCommand(i),i=  561,  640)/
0090        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0091        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0092        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0093        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0094        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0095        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0096        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0097        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0098        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0099        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0100        data (PenCommand(i),i=  641,  720)/
0101        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0102        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0103        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0104        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0105        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0106        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0107        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0108        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0109        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0110        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      /
0111        data (PenCommand(i),i=  721,  800)/
0112        . 1      , 0      , 1      , 1      , 1      , 1      , 0      , 1      ,
0113        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0114        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0115        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0116        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0117        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0118        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0119        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0120        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0121        . 1      , 0      , 1      , 1      , 1      , 1      , 0      , 1
0122        data (PenCommand(i),i=  801,  880)/
0123        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0124        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
```

171

```
0125        .  1      ,  1      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,
0126        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,
0127        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,  1      ,
0128        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,  1      ,
0129        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,
0130        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,
0131        .  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,  1      ,  1      ,
0132        .  1      ,  1      ,  0      ,  1      ,  1      ,  1      ,  1      ,  1      /
0133        data (PenCommand(i),i=  881,   960)/
0134        .  1      ,  1      ,  1      ,  0      ,  1      ,  0      ,  1      ,  1      ,
0135        .  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  1      ,
0136        .  1      ,  0      ,  1      ,  1      ,  1      ,  0      ,  1      ,  0      ,
0137        .  1      ,  0      ,  -      ,  0      ,  1      ,  1      ,  1      ,  1      ,
0138        .  1      ,  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0139        .  1      ,  1      ,  0      ,  1      ,  0      ,  1      ,  1      ,  1      ,
0140        .  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0141        .  1      ,  1      ,  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0142        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,  1      ,
0143        .  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      /
0144        data (PenCommand(i),i=  961,  1040)/
0145        .  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,
0146        .  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,
0147        .  1      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,
0148        .  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,
0149        .  0      ,  1      ,  0      ,  1      ,  1      ,  1      ,  0      ,  1      ,
0150        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,  1      ,
0151        .  0      ,  1      ,  0      ,  1      ,  1      ,  1      ,  0      ,  1      ,
0152        .  1      ,  1      ,  1      ,  1      ,  0      ,  1      ,  1      ,  1      ,
0153        .  0      ,  1      ,  0      ,  1      ,  1      ,  1      ,  1      ,  0      ,
0154        .  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      /
0155        data (PenCommand(i),i= 1041,  1120)/
0156        .  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  1      ,
0157        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0158        .  1      ,  1      ,  1      ,  1      ,  0      ,  1      ,  1      ,  1      ,
0159        .  1      ,  1      ,  1      ,  1      ,  0      ,  1      ,  1      ,  1      ,
0160        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,  1      ,
0161        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0162        .  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0163        .  1      ,  1      ,  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0164        .  1      ,  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0165        .  1      ,  0      ,  1      ,  .      ,  1      ,  1      ,  1      ,  1      /
0166        data (PenCommand(i),i= 1121,  1200)/
0167        .  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,  1      ,  1      ,
0168        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,
0169        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0170        .  1      ,  1      ,  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0171        .  1      ,  1      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,
0172        .  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,
0173        .  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,
0174        .  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,
0175        .  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,
0176        .  1      ,  1      ,  1      ,  1      ,  0      ,  1      ,  0      ,  1      /
0177        data (PenCommand(i),i= 1201,  1280)/
0178        .  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  0      ,
0179        .  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,
0180        .  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,  1      ,  0      ,
0181        .  1      ,  0      ,  1      ,  0      ,  1      ,  1      ,  1      ,  1      ,
0182        .  1      ,  1      ,  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0183        .  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0184        .  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0185        .  1      ,  1      ,  0      ,  1      ,         ,  1      ,  1      ,  1      ,
0186        .  1      ,  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,
0187        .  0      ,  1      ,  1      ,  1      ,  1      ,  1      ,  1      ,  0      /
0188        data (PenCommand(i),i= 1281,  1360)/
```

172

```
0189        . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 1      ,
0190        . 1      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0191        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0192        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0193        . 1      , 1      , 0      , 1      , 1      , 0      , 1      , 1      ,
0194        . 1      , 1      , 1      , 1      , 1      , 0      , 0      , 1      ,
0195        . 0      , 1      , 0      , 1      , 1      , 1      , 0      , 1      ,
0196        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0197        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0198        . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 0      /
0199        data (PenCommand(i),i= 1361, 1440)/
0200        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
0201        . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 1      ,
0202        . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 1      ,
0203        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 0      ,
0204        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0205        . 0      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
0206        . 0      , 1      , 0      , 1      , 1      , 1      , 0      , 1      ,
0207        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0208        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0209        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      /
0210        data (PenCommand(i),i= 1441, 1520)/
0211        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
0212        . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
0213        . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 0      ,
0214        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0215        . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 0      ,
0216        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0217        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0218        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0219        . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
0220        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      /
0221        data (PenCommand(i),i= 1521, 1600)/
0222        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
0223        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 0      ,
0224        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0225        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0226        . 1      , 0      , 1      , 1      , 1      , 1      , 0      , 1      ,
0227        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0228        . 1      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
0229        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0230        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0231        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
0232        data (PenCommand(i),i= 1601, 1680)/
0233        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0234        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0235        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0236        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0237        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0238        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0239        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0240        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0241        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0242        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0243        data (PenCommand(i),i= 1681, 1760)/
0244        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0245        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0246        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0247        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0248        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0249        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0250        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0251        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0252        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
```

```
0253        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0254        data (PenCommand(i),i= 1761, 1840)/
0255        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0256        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0257        . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 1      ,
0258        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0259        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 0      ,
0260        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0261        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0262        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0263        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0264        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      /
0265        data (PenCommand(i),i= 1841, 1920)/
0266        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0267        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0268        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0269        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0270        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0271        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0272        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0273        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0274        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0275        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      /
0276        data (PenCommand(i),i= 1921, 2000)/
0277        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0278        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0279        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0280        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0281        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0282        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0283        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0284        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0285        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0286        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0287        data (PenCommand(i),i= 2001, 2080)/
0288        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0289        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0290        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0291        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0292        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0293        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0294        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0295        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0296        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0297        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      /
0298        data (PenCommand(i),i= 2081, 2160)/
0299        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0300        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0301        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0302        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0303        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0304        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0305        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0306        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0307        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0308        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0309        data (PenCommand(i),i= 2161, 2240)/
0310        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0311        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0312        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0313        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0314        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0315        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0316        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
```

```
0317        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0318        . 1      , 1       , 1       , 0       , 1       , 1       , 1       , 1        ,
0319        . 1      , 1       , 1       , 1       , 1       , 0       , 1       , 1        /
0320        data (PenCommand(i),i= 2241, 2320)/
0321        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0322        . 0      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0323        . 1      , 1       , 1       , 0       , 1       , 1       , 1       , 1        ,
0324        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0325        . 0      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0326        . 0      , 1       , 1       , 0       , 1       , 1       , 1       , 1        ,
0327        . 0      , 1       , 1       , 1       , 1       , 0       , 1       , 1        ,
0328        . 1      , 0       , 1       , 1       , 1       , 0       , 1       , 0        ,
0329        . 1      , 0       , 1       , 1       , 1       , 0       , 1       , 0        ,
0330        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 0        /
0331        data (PenCommand(i),i= 2321,  2400)/
0332        . 1      , 0       , 1       , 1       , 1       , 1       , 1       , 1        ,
0333        . 1      , 1       , 0       , 1       , 0       , 1       , 1       , 1        ,
0334        . 0      , 1       , 0       , 1       , 0       , 1       , 1       , 1        ,
0335        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0336        . 1      , 1       , 0       , 1       , 1       , 0       , 1       , 1        ,
0337        . 1      , 1       , 1       , 0       , 1       , 0       , 1       , 0        ,
0338        . 1      , 0       , 1       , 0       , 1       , 0       , 1       , 1        ,
0339        . 1      , 0       , 1       , 1       , 1       , 0       , 1       , 1        ,
0340        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0341        . 0      , 1       , 0       , 1       , 1       , 1       , 1       , 1        /
0342        data (PenCommand(i),i= 2401,  2480)/
0343        . 0      , 1       , 0       , 1       , 1       , 1       , 0       , 1        ,
0344        . 0      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0345        . 1      , 0       , 1       , 1       , 0       , 1       , 1       , 1        ,
0346        . 1      , 1       , 1       , 0       , 1       , 0       , 1       , 0        ,
0347        . 1      , 0       , 1       , 1       , 0       , 1       , 0       , 1        ,
0348        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0349        . 1      , 1       , 0       , 1       , 0       , 1       , 1       , 1        ,
0350        . 0      , 1       , 0       , 1       , 0       , 1       , 0       , 1        ,
0351        . 0      , 1       , 0       , 1       , 0       , 1       , 0       , 1        ,
0352        . 0      , 1       , 0       , 1       , 0       , 1       , 0       , 1        /
0353        data (PenCommand(i),i= 2481,  2560)/
0354        . 0      , 1       , 0       , 1       , 0       , 1       , 0       , 1        ,
0355        . 0      , 1       , 0       , 1       , 0       , 1       , 1       , 1        ,
0356        . 1      , 1       , 0       , 1       , 1       , 1       , 1       , 1        ,
0357        . 1      , 0       , 1       , 1       , 1       , 1       , 1       , 0        ,
0358        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0359        . 1      , 1       , 0       , 1       , 1       , 1       , 1       , 1        ,
0360        . 0      , 1       , 1       , 1       , 1       , 1       , 0       , 1        ,
0361        . 1      , 1       , 1       , 1       , 0       , 1       , 1       , 1        ,
0362        . 1      , 1       , 1       , 1       , 1       , 0       , 1       , 1        ,
0363        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 1        /
0364        data (PenCommand(i),i= 2561,  2640)/
0365        . 1      , 1       , 1       , 0       , 1       , 1       , 1       , 1        ,
0366        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 0        ,
0367        . 1      , 0       , 1       , 0       , 1       , 0       , 1       , 0        ,
0368        . 1      , 0       , 1       , 0       , 1       , 0       , 1       , 0        ,
0369        . 1      , 0       , 1       , 0       , 1       , 0       , 1       , 0        ,
0370        . 1      , 0       , 1       , 0       , 1       , 0       , 1       , 0        ,
0371        . 1      , 0       , 1       , 0       , 1       , 0       , 1       , 0        ,
0372        . 1      , 0       , 1       , 0       , 1       , 0       , 1       , 0        ,
0373        . 1      , 0       , 1       , 0       , 1       , 0       , 1       , 0        ,
0374        . 1      , 0       , 1       , 0       , 1       , 0       , 1       , 0        ,
0375        data (PenCommand(i),i= 2641,  2720)/
0376        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0377        . 1      , 1       , 1       , 1       , 1       , 0       , 1       , 1        ,
0378        . 1      , 1       , 1       , 1       , 1       , 1       , 0       , 1        ,
0379        . 1      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
0380        . 0      , 1       , 1       , 1       , 1       , 1       , 1       , 1        ,
```

```
0381        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0382        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0383        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 0      ,
0384        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0385        . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 1      /
0386        data (PenCommand(i),i= 2721, 2800)/
0387        . 1      , 0      , 1      , 1      , 1      , 1      , 0      , 1      ,
0388        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 0      ,
0389        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0390        . 1      , 0      , 1      , 1      , 0      , 1      , 0      , 1      ,
0391        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0392        . 0      , 1      , 1      , 1      , 0      , 1      , 1      , 0      ,
0393        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0394        . 0      , 1      , 0      , 1      , 1      , 1      , 1      , 0      ,
0395        . 1      , 1      , 0      , 1      , 1      , 1      , 0      , 1      ,
0396        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      /
0397        data (PenCommand(i),i= 2801, 2880)/
0398        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0399        . 0      , 1      , 1      , 0      , 1      , 1      , 0      , 1      ,
0400        . 1      , 1      , 0      , 1      , 1      , 1      , 0      , 1      ,
0401        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0402        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0403        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      .
0404        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0405        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0406        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0407        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      /
0408        data (PenCommand(i),i= 2881, 2960)/
0409        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0410        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0411        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0412        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0413        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0414        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0415        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0416        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0417        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0418        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
0419        data (PenCommand(i),i= 2961, 3040)/
0420        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0421        . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0422        . 1      , 0      , 1      , 0      , 1      , 1      , 0      , 1      ,
0423        . 0      , 1      , 0      , 1      , 1      , 0      , 1      , 0      ,
0424        . 1      , 0      , 1      , 1      , 0      , 1      , 1      , 1      ,
0425        . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0426        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0427        . 1      , 1      , 0      , 1      , 1      , 1      , 0      , 1      ,
0428        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0429        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
0430        data (PenCommand(i),i= 3041, 3120)/
0431        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0432        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0433        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0434        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0435        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0436        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0437        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0438        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0439        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0440        . 1      , 0      , 1      , 1      , 0      , 1      , 1      , 0      ,
0441        data (PenCommand(i),i= 3121, 3200)
0442        . 0      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
0443        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0444        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      .
```

176

```
0445      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0446      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0447      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0448      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0449      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0450      . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
0451      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      /
0452      data (PenCommand(i),i= 3201, 3280)/
0453      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0454      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
0455      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0456      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0457      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
0458      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0459      . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0460      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0461      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0462      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
0463      data (PenCommand(i),i= 3281, 3360)/
0464      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0465      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0466      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0467      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0468      . 0      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
0469      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0470      . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
0471      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0472      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0473      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0474      data (PenCommand(i),i= 3361, 3440)/
0475      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0476      . 1      , 0      , 1      , 1      , 0      , 1      , 1      , 1      ,
0477      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0478      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0479      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0480      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0481      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0482      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0483      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0484      . 1      , 1      , 0      , 1      , 1      , 0      , 1      , 1      /
0485      data (PenCommand(i),i= 3441, 3520)/
0486      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0487      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0488      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0489      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0490      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0491      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0492      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0493      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0494      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0495      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      /
0496      data (PenCommand(i),i= 3521, 3600)/
0497      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0498      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0499      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0500      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0501      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0502      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0503      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0504      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0505      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0506      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0507      data (PenCommand(i),i= 3601, 3680)/
0508      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
```

177

```
0509        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0510        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0511        . 1      , 1      , 1       , 0      , 1      , 1      , 1      , 1      ,
0512        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0513        . 1      , 0      , 1       , 1      , 1      , 1      , 1      , 1      ,
0514        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0515        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0516        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 0      ,
0517        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      /
0518        data (PenCommand(i),i= 3681, 3760)/
0519        . 1      , 1      , 0       , 1      , 1      , 1      , 1      , 1      ,
0520        . 1      , 1      , 1       , 0      , 1      , 1      , 1      , 1      ,
0521        . 1      , 1      , 1       , 1      , 1      , 0      , 1      , 1      ,
0522        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0523        . 1      , 1      , 0       , 1      , 1      , 1      , 1      , 1      ,
0524        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0525        . 1      , 1      , 0       , 1      , 1      , 1      , 1      , 1      ,
0526        . 1      , 1      , 1       , 0      , 1      , 1      , 1      , 1      ,
0527        . 1      , 1      , 1       , 1      , 1      , 1      , 0      , 1      ,
0528        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      /
0529        data (PenCommand(i),i= 3761, 3840)/
0530        . 1      , 1      , 1       , 0      , 1      , 1      , 1      , 1      ,
0531        . 1      , 1      , 1       , 1      , 1      , 1      , 0      , 1      ,
0532        . 1      , 1      , 1       , 0      , 1      , 1      , 1      , 0      ,
0533        . 1      , 1      , 1       , 1      , 1      , 0      , 1      , 1      ,
0534        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0535        . 1      , 1      , 1       , 1      , 0      , 1      , 1      , 1      ,
0536        . 1      , 1      , 1       , 1      , 0      , 1      , 1      , 1      ,
0537        . 1      , 1      , 1       , 1      , 1      , 0      , 1      , 1      ,
0538        . 0      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0539        . 1      , 1      , 1       , 0      , 1      , 1      , 1      , 1      /
0540        data (PenCommand(i),i= 3841, 3920)/
0541        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0542        . 1      , 1      , 1       , 1      , 0      , 1      , 1      , 1      ,
0543        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0544        . 1      , 1      , 1       , 1      , 0      , 1      , 1      , 1      ,
0545        . 1      , 1      , 1       , 1      , 0      , 1      , 1      , 1      ,
0546        . 1      , 1      , 0       , 1      , 1      , 0      , 1      , 1      ,
0547        . 1      , 0      , 1       , 0      , 1      , 0      , 1      , 0      ,
0548        . 1      , 0      , 1       , 0      , 1      , 0      , 1      , 1      ,
0549        . 1      , 0      , 1       , 1      , 1      , 1      , 1      , 0      ,
0550        . 1      , 0      , 1       , 0      , 1      , 0      , 1      , 0      /
0551        data (PenCommand(i),i= 3921, 4000)/
0552        . 1      , 0      , 1       , 1      , 1      , 0      , 1      , 1      ,
0553        . 1      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0554        . 0      , 1      , 1       , 1      , 1      , 1      , 1      , 1      ,
0555        . 1      , 0      , 1       , 0      , 1      , 0      , 1      , 0      ,
0556        . 1      , 0      , 1       , 1      , 1      , 1      , 0      , 1      ,
0557        . 0      , 1      , 0       , 1      , 0      , 1      , 1      , 1      ,
0558        . 1      , 1      , 0       , 1      , 0      , 1      , 0      , 1      ,
0559        . 0      , 1      , 0       , 1      , 0      , 1      , 0      , 1      ,
0560        . 0      , 1      , 0       , 1      , 0      , 1      , 0      , 1      ,
0561        . 0      , 1      , 0       , 1      , 0      , 1      , 0      , 1      /
0562        data (PenCommand(i),i= 4001, 4080)/
0563        . 0      , 1      , 0       , 1      , 0      , 1      , 0      , 1      ,
0564        . 1      , 1      , 1       , 1      , 0      , 1      , 0      , 1      ,
0565        . 0      , 1      , 0       , 1      , 0      , 1      , 0      , 1      ,
0566        . 0      , 1      , 1       , 1      , 0      , 1      , 0      , 1      ,
0567        . 0      , 1      , 0       , 1      , 1      , 1      , 0      , 1      ,
0568        . 0      , 1      , 0       , 1      , 0      , 1      , 0      , 1      ,
0569        . 0      , 1      , 0       , 1      , 0      , 1      , 0      , 1      ,
0570        . 0      , 1      , 0       , 1      , 0      , 1      , 0      , 1      ,
0571        . 0      , 1      , 0       , 1      , 0      , 1      , 0      , 1      ,
0572        . 0      , 1      , 0       , 1      , 0      , 1      , 0      , 1      /
```

```
0573          data (PenCommand(i),i= 4081, 4160)/
0574     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0575     . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0576     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0577     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0578     . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0579     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0580     . 0      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0581     . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0582     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0583     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
0584          data (PenCommand(i),i= 4161, 4240)/
0585     . 0      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0586     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0587     . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 0      ,
0588     . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 1      ,
0589     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0590     . 0      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0591     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0592     . 0      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0593     . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
0594     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      /
0595          data (PenCommand(i),i= 4241, 4320)/
0596     . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 1      ,
0597     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0598     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0599     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0600     . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0601     . 1      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0602     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0603     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0604     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0605     . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 0      /
0606          data (PenCommand(i),i= 4321, 4400)/
0607     . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0608     . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0609     . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0610     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0611     . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0612     . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0613     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0614     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0615     . 0      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0616     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0617          data (PenCommand(i),i= 4401, 4480)/
0618     . 1      , 0      , 1      , 1      , 1      , 1      , 0      , 1      ,
0619     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0620     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0621     . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 0      ,
0622     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0623     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0624     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0625     . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0626     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0627     . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1
0628          data (PenCommand(i),i= 4481, 4560)/
0629     . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0630     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0631     . 1      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0632     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0633     . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0634     . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0635     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0636     . 1      , 0      , 1      , 1      , 0      , 1      , 1      , 0      ,
```

179

```
0637        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0638        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0639        data (PenCommand(i),i= 4561, 4640)/
0640        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0641        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0642        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0643        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0644        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0645        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0646        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0647        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0648        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0649        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      /
0650        data (PenCommand(i),i= 4641, 4720)/
0651        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0652        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0653        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0654        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0655        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0656        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0657        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0658        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0659        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0660        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0661        data (PenCommand(i),i= 4721, 4800)/
0662        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0663        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0664        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0665        . 1      , 1      , 1      , 0      , 1      , 1      , 0      , 1      ,
0666        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0667        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0668        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0669        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0670        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0671        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0672        data (PenCommand(i),i= 4801, 4880)/
0673        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0674        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0675        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0676        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0677        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0678        . 1      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
0679        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0680        . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 0      ,
0681        . 1      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0682        . 0      , 1      , 1      , 1      , 1      , 1      , 0      , 1      /
0683        data (PenCommand(i),i= 4881, 4960)/
0684        . 0      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
0685        . 0      , 1      , 0      , 1      , 1      , 1      , 0      , 1      ,
0686        . 0      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
0687        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0688        . 1      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
0689        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0690        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0691        . 0      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0692        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 0      ,
0693        . 1      , 0      , 1      , 1      , 1      , 1      , 0      , 1      ,
0694        data (PenCommand(i),i= 4961, 5040)/
0695        . 1      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0696        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0697        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0698        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0699        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
0700        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
```

```
0701        . 1      , 1      , 1       1      , 0      , 1      , 1      , 1      ,
0702        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0703        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0704        . 1      , 1      , 1      , 1      , 0      , 1      , 1      ,        /
0705        data (PenCommand(i),i= 5041, 5120)/
0706        . 0      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0707        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0708        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0709        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0710        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0711        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0712        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 0      ,
0713        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0714        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0715        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      /
0716        data (PenCommand(i),i= 5121, 5200)/
0717        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0718        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0719        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0720        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0721        . 0      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0722        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0723        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0724        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0725        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0726        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0727        data (PenCommand(i),i= 5201, 5280)/
0728        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0729        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0730        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0731        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0732        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0733        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0734        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0735        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0736        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0737        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      /
0738        data (PenCommand(i),i= 5281, 5360)/
0739        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0740        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0741        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0742        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0743        . 0      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0744        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0745        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0746        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0747        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0748        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0749        data (PenCommand(i),i= 5361, 5440)/
0750        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 0      ,
0751        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0752        . 1      , 0      , 1      , 1      , 0      , 1      , 0      , 1      ,
0753        . 0      , 1      , 0      , 1      , 1      , 0      , 1      , 1      ,
0754        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0755        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0756        . 0      , 1      ,        ,   1    ,        ,        ,        ,        ,
0757        . 1      , 0      , 1      , 0      , 1      , 1      ,        ,        ,
0758        . 1      , 1      , 1      , 0      , 1      ,        , 1      ,        ,
0759        . 1      , 1      , 1      ,   /    , 1      , 1      , 1      ,        ,
0760        data (PenCommand(i),i= 5441, 5520)/
0761        . 1      , 1      ,        , 1      , 1      , 1      , 1      , 0      ,
0762        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0763        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0764        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
```

```
0765      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0766      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0767      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0768      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0769      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0770      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      /
0771      data (PenCommand(i),i= 5521,  5600)/
0772      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0773      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0774      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0775      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0776      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0777      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0778      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0779      . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
0780      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0781      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 0      /
0782      data (PenCommand(i),i= 5601,  5680)/
0783      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0784      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0785      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0786      . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0787      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0788      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0789      . 0      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
0790      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0791      . 1      , 1      , 0      , 1      , 1      , 1      , 0      , 1      ,
0792      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
0793      data (PenCommand(i),i= 5681,  5760)/
0794      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0795      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0796      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0797      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0798      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0799      . 0      , 1      , 0      , 1      , 0      , 1      , 1      , 0      ,
0800      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0801      . 1      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0802      . 0      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
0803      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 0      /
0804      data (PenCommand(i),i= 5761,  5840)/
0805      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0806      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0807      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
0808      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0809      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0810      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0811      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0812      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0813      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0814      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      /
0815      data (PenCommand(i),i= 5841,  5920)/
0816      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0817      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0818      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0819      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 0      ,
0820      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0821      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0822      . 0      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0823      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0824      . 0      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0825      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0826      data (PenCommand(i),i= 5921,  6000)/
0827      . 0      , 1      , 0      , 1      , 0      , 1      , 1      , 0      ,
0828      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
```

```
0829        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0830        . 1      , 0      , 1      , 1      , 0      , 1      , 0      , 1      ,
0831        . 0      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0832        . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0833        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0834        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0835        . 0      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0836        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
0837        data (PenCommand(i),i= 6001, 6080)/
0838        . 0      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0839        . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0840        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0841        . 0      , 1      , C      , 1      , 0      , 1      , 0      , 1      ,
0842        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0843        . 0      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0844        . 0      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0845        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0846        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
0847        . 0      , 1      , 1      , 1      , 1      , 1      , 0      , 1      /
0848        data (PenCommand(i),i= 6081, 6160)/
0849        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0850        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0851        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0852        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0853        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0854        . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 0      ,
0855        . 1      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
0856        . 1      , 0      , 1      , 1      , 0      , 1      , 1      , 1      ,
0857        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0858        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      /
0859        data (PenCommand(i),i= 6161, 6240)/
0860        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0861        . 1      , 1      , 0      , 1      , 1      , 1      , 0      , 1      ,
0862        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0863        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0864        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0865        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0866        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0867        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0868        . 0      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0869        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      /
0870        data (PenCommand(i),i= 6241, 6320)/
0871        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0872        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0873        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0874        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0875        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0876        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0877        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0878        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0879        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0880        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0881        data (PenCommand(i),i= 6321, 6400)/
0882        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0883        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0884        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0885        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0886        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0887        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0888        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0889        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0890        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0891        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
0892        data (PenCommand(i),i= 6401, 6480)/
```

```
0893        . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 1      ,
0894        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0895        . 0      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0896        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0897        . 0      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0898        . 1      , 0      , 1      , 1      , 1      , 1      , 0      , 1      ,
0899        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0900        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0901        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0902        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      /
0903        data (PenCommand(i),i= 6481, 6560)/
0904        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0905        . 1      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
0906        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0907        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 0      ,
0908        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
0909        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0910        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0911        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0912        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0913        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      /
0914        data (PenCommand(i),i= 6561, 6640)/
0915        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0916        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0917        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0918        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0919        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0920        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
0921        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0922        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0923        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
0924        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      /
0925        data (PenCommand(i),i= 6641, 6720)/
0926        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0927        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
0928        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0929        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0930        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
0931        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
0932        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 0      ,
0933        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
0934        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
0935        . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      /
0936        data (PenCommand(i),i= 6721, 6800)/
0937        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 0      ,
0938        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0939        . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 0      ,
0940        . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 0      ,
0941        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0942        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0943        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0944        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0945        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0946        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      /
0947        data (PenCommand(i),i= 6801, 6880)/
0948        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0949        . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
0950        . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 0      ,
0951        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0952        . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 0      ,
0953        . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
0954        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
0955        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
0956        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
```

```
0957        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1      /
0958        data (PenCommand(i),i= 6881, 6960)/
0959        . 1      , 1      , 1       , 1      , 1       . 0      , 1      , 1      ,
0960        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
0961        . 1      , 1      , 1       , 1      , 1       , 1      , 0      , 1      ,
0962        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
0963        . 1      , 1      , 1       , 0      , 1       , 1      , 1      , 1      ,
0964        . 1      , 1      , 1       , 1      , 0       , 1      , 1      , 1      ,
0965        . 1      , 1      , 1       , 1      , 1       , 1      , 0      , 1      ,
0966        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
0967        . 1      , 1      , 1       , 1      , 0       , 1      , 0      , 1      ,
0968        . 0      , 1      , 0       , 1      , 0       , 1      , 0      , 1      /
0969        data (PenCommand(i),i= 6961, 7040)/
0970        . 0      , 1      , 0       , 1      , 1       , 1      , 1      , 1      ,
0971        . 1      , 0      , 1       , 1      , 1       , 1      , 1      , 1      ,
0972        . 1      , 1      , 0       , 1      , 1       , 1      , 1      , 1      ,
0973        . 1      , 1      , 1       , 1      , 0       , 1      , 1      , 1      ,
0974        . 1      , 1      , 1       , 1      , 1       , 1      , 0      , 1      ,
0975        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
0976        . 1      , 0      , 1       , 1      , 1       , 1      , 1      , 1      ,
0977        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
0978        . 1      , 1      , 0       , 1      , 1       , 1      , 1      , 1      ,
0979        . 0      , 1      , 1       , 1      , 1       , 1      , 0      , 1      /
0980        data (PenCommand(i),i= 7041, 7120)/
0981        . 1      , 1      , 1       , 1      , 0       , 1      , 1      , 1      ,
0982        . 1      , 1      , 1       , 1      , 1       , 1      , 0      , 1      ,
0983        . 1      , 1      , 1       , 1      , 1       , 1      , 0      , 1      ,
0984        . 1      , 1      , 1       , 0      , 1       , 0      , 1      , 1      ,
0985        . 1      , 1      , 1       , 1      , 1       , 1      , 0      , 1      ,
0986        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
0987        . 0      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
0988        . 1      , 1      , 1       , 1      , 1       , 0      , 1      , 1      ,
0989        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
0990        . 0      , 1      , 1       , 1      , 1       , 1      , 1      , 1      /
0991        data (PenCommand(i),i= 7121, 7200)/
0992        . 1      , 0      , 1       , 1      , 1       , 1      , 1      , 1      ,
0993        . 1      , 1      , 1       , 0      , 1       , 1      , 0      , 1      ,
0994        . 0      , 1      , 1       , 1      , 1       , 0      , 1      , 1      ,
0995        . 1      , 1      , 0       , 1      , 0       , 1      , 1      , 1      ,
0996        . 0      , 1      , 0       , 1      , 1       , 1      , 1      , 0      ,
0997        . 1      , 1      , 1       , 0      , 1       , 1      , 1      , 0      .
0998        . 1      , 0      , 1       , 0      , 1       , 1      , 1      , 0      ,
0999        . 1      , 0      , 1       , 1      , 1       , 1      , 1      , 1      ,
1000        . 1      , 1      , 1       , 0      , 1       , 1      , 1      , 1      ,
1001        . 1      , 1      , 1       , 1      , 1       , 0      , 1      , 1      /
1002        data (PenCommand(i),i= 7201, 7280)/
1003        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
1004        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 0      ,
1005        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
1006        . 1      , 1      , 1       , 1      , 1       , 1      , 0      , 1      ,
1007        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
1008        . 1      , 1      , 1       , 1      , 1       , 0      , 1      , 1      ,
1009        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 0      ,
1010        . 1      , 1      , 0       , 1      , 1       , 1      , 1      , 1      ,
1011        . 1      , 1      , 1       , 1      , 1       , 0      , 1      , 1      ,
1012        . 1      , 1      , 1       , 1      , 1       , 1      , 1      , 1
1013        data (PenCommand(i),i= 7281, 7360)
1014        . 1      , 1      , 1       , 1      , 1       , 0      , 1      , 1      ,
1015        . 0      , 1      , 1       , 1      , 1       , 1      , 1      , 1      ,
1016        . 1      , 1      , 1       , 1      , 0       , 1      , 1      , 1      ,
1017        . 1      , 1      , 1       , 0      , 1       , 1      , 1      , 1      ,
1018        . 0      , 1      , 1       , 0      , 1       , 1      , 1      , 1      ,
1019        . 1      , 1      , 1       , 1      , 0       , 1      , 1      , 1      ,
1020        . 1      , 1      , 1       , 1      , 1       , 1      , 0      , 1      .
```

```
1021      . 0    , 1    , 0    , 1    , 0    , 1    , 0    , 1    ,
1022      . 0    , 1    , 0    , 1    , 0    , 1    , 0    , 1    ,
1023      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 1    /
1024      data (PenCommand(i),i= 7361, 7440)/
1025      . 0    , 1    , 0    , 1    , 1    , 1    , 1    , 0    ,
1026      . 1    , 0    , 1    , 1    , 1    , 0    , 1    , 0    ,
1027      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1028      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1029      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1030      . 1    , 0    , 1    , 1    , 1    , 0    , 1    , 0    ,
1031      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1032      . 1    , 0    , 1    , 1    , 1    , 1    , 1    , 1    ,
1033      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1034      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 1    /
1035      data (PenCommand(i),i= 7441, 7520)/
1036      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1037      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1038      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 1    ,
1039      . 0    , 1    , 1    , 1    , 0    , 1    , 0    , 1    ,
1040      . 0    , 1    , 0    , 1    , 0    , 1    , 0    , 1    ,
1041      . 0    , 1    , 1    , 1    , 1    , 1    , 0    , 1    ,
1042      . 1    , 1    , 1    , 0    , 1    , 0    , 1    , 0    ,
1043      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1044      . 1    , 1    , 1    , 1    , 1    , 0    , 1    , 0    ,
1045      . 1    , 1    , 1    , 1    , 1    , 0    , 1    , 0    /
1046      data (PenCommand(i),i= 7521, 7600)/
1047      . 1    , 0    , 1    , 1    , 1    , 1    , 1    , 1    ,
1048      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1049      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1050      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1051      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1052      . 1    , 0    , 1    , 0    , 1    , 1    , 0    , 1    ,
1053      . 0    , 1    , 0    , 1    , 0    , 1    , 1    , 1    ,
1054      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1055      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1056      . 1    , 1    , 0    , 1    , 0    , 1    , 1    , 1    /
1057      data (PenCommand(i),i= 7601  7680)/
1058      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1059      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1060      . 1    , 1    , 1    , 0    , 1    , 1    , 1    , 0    ,
1061      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1062      . 1    , 1    , 1    , 0    , 1    , 1    , 1    , 1    ,
1063      . 1    , 1    , 1    , 1    , 1    , 0    , 1    , 1    ,
1064      . 1    , 1    , 0    , 1    , 0    , 1    , 0    , 1    ,
1065      . 0    , 1    , 0    , 1    , 1    , 1    , 1    , 1    ,
1066      . 1    , 1    , 1    , 1    , 0    , 1    , 1    , 1    ,
1067      . 1    , 1    , 1    , 1    , 0    , 1    , 1    , 1    /
1068      data (PenCommand(i),i= 7681, 7760)/
1069      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1070      . 0    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1071      . 1    , 1    , 1    , 1    , 0    , 1    , 1    , 1    ,
1072      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 0    ,
1073      . 1    , 0    , 1    , 0    , 1    , 1    , 1    , 1    ,
1074      . 1    , 1    , 0    , 1    , 1    , 1    , 1    , 1    ,
1075      . 1    , 0    , 1    , 1    , 1    , 1    , 1    , 1    ,
1076      . 1    , 1    , 1    , 1    , 0    , 1    , 1    , 1    ,
1077      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1078      . 1    , 0    , 1    , 1    , 1    , 1    , 1    , 1    ,
1079      data (PenCommand(i),i= 7761, 7840)
1080      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 0    ,
1081      . 1    , 1    , 1    , 1    , 0    , 1    , 1    , 1    ,
1082      . 1    , 0    , 1    , 1    , 1    , 1    , 1    , 0    ,
1083      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1084      . 0    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
```

```
1085      . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 1      ,
1086      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1087      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1088      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1089      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
1090      data (PenCommand(i),i= 7841, 7920)/
1091      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1092      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
1093      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1094      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1095      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 0      ,
1096      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1097      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1098      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
1099      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1100      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      /
1101      data (PenCommand(i),i= 7921, 8000)/
1102      . 0      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
1103      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 0      ,
1104      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1105      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
1106      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
1107      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1108      . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 1      ,
1109      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1110      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1111      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 0      /
1112      data (PenCommand(i),i= 8001, 8080)/
1113      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
1114      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1115      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
1116      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1117      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1118      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1119      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
1120      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
1121      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1122      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      /
1123      data (PenCommand(i),i= 8081, 8160)/
1124      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
1125      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1126      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
1127      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1128      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1129      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
1130      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
1131      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1132      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1133      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
1134      data (PenCommand(i),i= 8161, 8240)/
1135      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1136      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1137      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1138      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1139      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1140      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1141      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1142      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1143      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1144      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1
1145      data (PenCommand(i),i= 8241, 8320)/
1146      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1147      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1148      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
```

187

```
1149      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1150      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1151      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
1152      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1153      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1154      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1155      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
1156          data (PenCommand(i),i= 8321,  8400)/
1157      . 1      , 1      , 1      , 1      . 1      , 1      , 1      , 1      ,
1158      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
1159      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1160      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1161      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1162      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1163      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1164      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1165      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1166      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
1167          data (PenCommand(i),i= 8401,  8480)/
1168      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1169      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1170      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
1171      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1172      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1173      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1174      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1175      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1176      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1177      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
1178          data (PenCommand(i),i= 8481,  8560)/
1179      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1180      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1181      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1182      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1183      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1184      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
1185      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1186      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
1187      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1188      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      /
1189          data (PenCommand(i),i= 8561,  8640)/
1190      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1191      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1192      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1193      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1194      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1195      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1196      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1197      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1198      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 0      ,
1199      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      /
1200          data (PenCommand(i),i= 8641,  8720)/
1201      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1202      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1203      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1204      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1205      . 1      , 1      , 0      , 1      , 1      , 0      , 1      , 0      ,
1206      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
1207      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1208      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
1209      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1210      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      /
1211          data (PenCommand(i),i= 8721,  8800)/
1212      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
```

```
1213        . 1      . 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1214        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1215        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1216        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1217        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1218        . 0      , 1      , 0      , 1      , 0      . 1      , 0      , 1      ,
1219        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1220        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1221        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
1222        data (PenCommand(i),i= 8801, 8880)/
1223        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1224        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1225        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1226        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1227        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1228        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1229        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1230        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1231        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1232        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
1233        data (PenCommand(i),i= 8881, 8960)/
1234        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1235        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1236        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1237        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1238        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1239        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1240        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1241        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1242        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1243        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
1244        data (PenCommand(i),i= 8961, 9040)/
1245        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1246        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1247        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1248        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1249        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1250        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1251        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1252        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1253        . 0      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
1254        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
1255        data (PenCommand(i),i= 9041, 9120)/
1256        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1257        . 1      , 1      , 1      , 1      , 0      , 1      , 0      , 1      ,
1258        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1259        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1260        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1261        . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1262        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1263        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1264        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1265        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
1266        data (PenCommand(i),i= 9121, 9200)/
1267        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1268        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1269        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1270        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1271        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1272        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1273        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1274        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1275        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1276        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      /
```

189

```
1277            data (PenCommand(i),i= 9201, 9280)/
1278     . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1279     . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1280     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1281     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1282     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1283     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1284     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1285     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1286     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1287     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1288            data (PenCommand(i),i= 9281, 9360)/
1289     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1290     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1291     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1292     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1293     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1294     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1295     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1296     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1297     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1298     . 0      , 1      , 0      , 1      , 1      , 1      , 1      , 1      , /
1299            data (PenCommand(i),i= 9361, 9440)/
1300     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1301     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1302     . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1303     . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1304     . 0      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
1305     . 0      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1306     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1307     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1308     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      , /
1309     . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      , /
1310            data (PenCommand(i),i= 9441, 9520)/
1311     . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1312     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1313     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1314     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1315     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1316     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1317     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1318     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1319     . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      , /
1320     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      , /
1321            data (PenCommand(i),i= 9521, 9600)/
1322     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1323     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1324     . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
1325     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1326     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1327     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
1328     . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 1      ,
1329     . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 0      ,
1330     . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 0      ,
1331     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1332            data (PenCommand(i),i= 9601, 9680)
1333     . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
1334     . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 0      ,
1335     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1336     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1337     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1338     . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1339     . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1340     . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
```

```
1341      . 1       , 1       , 1       , 1       , 1       , 1       , 1       , 1       .
1342      . 1       , 1       , 1       , 1       , 1       , 1       , 1       , 1
1343      data (PenCommand(i),i= 9681, 9760)/
1344      . 1       , 1       , 1       , 1       , 1       , 1       , 1       , 1       .
1345      . 1       , 1       , 0       , 1       , 0       , 1       , 0       , 1       .
1346      . 1       , 1       , 1       , 1       , 1       , 0       , 1       , 0       .
1347      . 1       , 1       , 1       , 0       , 1       , 0       , 1       , 0       .
1348      . 1       , 1       , 1       , 1       , 1       , 1       , 1       , 0       .
1349      . 1       , 1       , 0       , 1       , 0       , 1       , 0       , 1       .
1350      . 0       , 1       , 0       , 1       , 0       , 1       , 0       , 1       .
1351      . 0       , 1       , 0       , 1       , 0       , 1       , 1       , 1       .
1352      . 0       , 1       , 0       , 1       , 1       , 0       , 1       , 1       .
1353      . 1       , 0       , 1       , 1       , 1       , 0       , 1       , 1       .
1354      data (PenCommand(i),i= 9761, 9840)/
1355      . 1       , 1       , 1       , 1       , 1       , 0       , 1       , 1       .
1356      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       .
1357      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 1       .
1358      . 1       , 1       , 1       , 0       , 1       , 1       , 1       , 1       .
1359      . 1       , 1       , 0       , 1       , 0       , 1       , 1       , 1       .
1360      . 1       , 1       , 1       , 1       , 1       , 1       , 1       , 1       .
1361      . 1       , 1       , 1       , 0       , 1       , 0       , 1       , 1       .
1362      . 1       , 0       , 1       , 0       , 1       , 1       , 1       , 0       .
1363      . 1       , 0       , 1       , 1       , 1       , 1       , 1       , 1       .
1364      . 0       , 1       , 1       , 1       , 1       , 1       , 1       , 1       /
1365      data (PenCommand(i),i= 9841, 9920)/
1366      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       .
1367      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       .
1368      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 1       .
1369      . 1       , 1       , 1       , 1       , 1       , 1       , 0       , 1       .
1370      . 1       , 1       , 1       , 1       , 1       , 1       , 1       , 0       .
1371      . 1       , 1       , 1       , 1       , 1       , 1       , 1       , 0       .
1372      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 1       .
1373      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       .
1374      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       .
1375      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       /
1376      data (PenCommand(i),i= 9921,10000)/
1377      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       .
1378      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       .
1379      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       .
1380      . 1       , 0       , 1       , 1       , 1       , 0       , 1       , 1       .
1381      . 1       , 0       , 1       , 0       , 1       , 1       , 1       , 1       .
1382      . 1       , 0       , 1       , 0       , 1       , 1       , 1       , 1       .
1383      . 1       , 1       , 1       , 1       , 0       , 1       , 1       , 1       .
1384      . 1       , 1       , 1       , 1       , 1       , 0       , 1       , 1       .
1385      . 1       , 1       , 1       , 1       , 1       , 1       , 0       , 1       .
1386      . 1       , 1       , 1       , 1       , 1       , 1       , 1       , 1       /
1387      data (PenCommand(i),i=10001,10080)/
1388      . 0       , 1       , 1       , 1       , 1       , 1       , 1       , 1       .
1389      . 0       , 1       , 0       , 1       , 0       , 1       , 0       , 1       .
1390      . 0       , 1       , 1       , 1       , 0       , 1       , 1       , 1       .
1391      . 0       , 1       , 1       , 1       , 0       , 1       , 1       , 1       .
1392      . 0       , 1       , 0       , 1       , 0       , 1       , 0       , 1       .
1393      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       .
1394      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       .
1395      . 1       , 0       , 1       , 0       , 1       , 0       , 1       , 0       .
1396      . 1       , 0       , 1       , 1       , 0       , 1       , 0       , 1       .
1397      . 1       , 1       , 0       , 1       , 0       , 1       , 0       , 1       .
1398      data (PenCommand(i),i=10081,10160)/
1399      . 0       , 1       , 0       , 1       , 0       , 1       , 1       , 1       .
1400      . 1       , 1       , 1       , 1       , 1       , 1       , 1       , 1       .
1401      . 1       , 1       , 1       , 1       , 1       , 1       , 0       , 1       .
1402      . 0       , 1       , 1       , 1       , 1       , 1       , 0       , 1       .
1403      . 1       , 1       , 1       , 0       , 1       , 1       , 1       , 1       .
1404      . 1       , 1       , 1       , 1       , 1       , 1       , 1       , 1       .
```

191

```
1405      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1406      . 1      . 1      . 1      . 0      . 1      . 1      . 1      . 1
1407      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1408      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1409          data (PenCommand(i),i=10161,10240)/
1410      . 0      . 1      . 1      . 1      . 1      . 1      . 1      . 0
1411      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1412      . 1      . 1      . 1      . 1      . 0      . 1      . 1      . 1
1413      . 1      . 1      . 1      . 1      . 0      . 1      . 1      . 1
1414      . 1      . 1      . 1      . 1      . 1      . 1      . 0      . 1
1415      . 0      . 1      . 0      . 1      . 1      . 0      . 1      . 1
1416      . 1      . 1      . 1      . 1      . 0      . 1      . 1      . 1
1417      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1418      . 1      . 1      . 1      . 0      . 1      . 1      . 1      . 1
1419      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1420          data (PenCommand(i),i=10241,10320)/
1421      . 1      . 0      . 1      . 1      . 1      . 1      . 1      . 1
1422      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1423      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 0
1424      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1425      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1426      . 1      . 1      . 1      . 1      . 0      . 1      . 1      . 1
1427      . 1      . 1      . 1      . 1      . 1      . 1      . 0      . 1
1428      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1429      . 0      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1430      . 1      . 1      . 1      . 0      . 1      . 1      . 1      . 1
1431          data (PenCommand(i),i=10321,10400)/
1432      . 1      . 1      . 1      . 1      . 1      . 0      . 1      . 1
1433      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1434      . 0      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1435      . 1      . 0      . 1      . 1      . 1      . 1      . 1      . 1
1436      . 1      . 1      . 0      . 1      . 1      . 1      . 1      . 1
1437      . 1      . 1      . 0      . 1      . 1      . 1      . 1      . 1
1438      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1439      . 1      . 1      . 1      . 1      . 0      . 1      . 1      . 1
1440      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 0
1441      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1442          data (PenCommand(i),i=10401,10480)/
1443      . 1      . 1      . 1      . 1      . 0      . 1      . 1      . 1
1444      . 1      . 1      . 1      . 0      . 1      . 1      . 1      . 1
1445      . 1      . 1      . 1      . 1      . 1      . 0      . 1      . 1
1446      . 1      . 1      . 1      . 1      . 1      . 0      . 1      . 1
1447      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1448      . 1      . 1      . 1      . 0      . 1      . 1      . 1      . 1
1449      . 1      . 1      . 1      . 1      . 1      . 1      . 0      . 1
1450      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1451      . 1      . 1      . 0      . 1      . 1      . 1      . 1      . 1
1452      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1453          data (PenCommand(i),i=10481,10560)/
1454      . 0      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1455      . 1      . 1      . 1      . 1      . 1      . 1      . 0      . 1
1456      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1457      . 1      . 1      . 0      . 1      . 1      . 1      . 1      . 1
1458      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1459      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1460      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1461      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1462      . 1      . 1      . 1      . 1      . 0      . 1      . 1
1463      . 1      . 1      . 1      . 1      . 1      . 1      . 0
1464          data (PenCommand(i),i=10561,10640)/
1465      . 1      . 1      . 1      . 1      . 1      . 1      . 1      . 1
1466      . 1      . 1      . 0      . 1      . 1      . 1      . 1      . 1
1467      . 1      . 1      . 1      . 0      . 1      . 1      . 1      . 0
1468      . 1      . 1      . 1      . 1      . 1      . 1      . 1
```

```
1469        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1470        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
1471        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1472        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
1473        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1474        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1475        data (PenCommand(i),i=10641,10720)/
1476        . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1477        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1478        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1479        . 1      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
1480        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1481        . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1482        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1483        . 0      , 1      , 0      , 1      , 1      , 0      , 1      , 1      ,
1484        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1485        . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1486        data (PenCommand(i),i=10721,10800)/
1487        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1488        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1489        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1490        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1491        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1492        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1493        . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
1494        . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
1495        . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
1496        . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
1497        data (PenCommand(i),i=10801,10880)/
1498        . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
1499        . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 1      ,
1500        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1501        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1502        . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
1503        . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1504        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1505        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1506        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1507        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1508        data (PenCommand(i),i=10881,10960)/
1509        . 0      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
1510        . 0      , 1      , 0      , 1      , 1      , 1      , 0      , 1      ,
1511        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1512        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1513        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1514        . 0      , 1      , 0      , 1      , 1      , 1      , 0      , 1      ,
1515        . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1516        . 0      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
1517        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1518        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1519        data (PenCommand(i),i=10961,11040)/
1520        . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
1521        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1522        . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1523        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
1524        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
1525        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1526        . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 0      ,
1527        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1528        . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1529        . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1530        data (PenCommand(i),i=11041,11120)/
1531        . 0      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1532        . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 1      ,
```

```
1533      . 0    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1334      . 1    , 1    , 1    , 1    , 1    , 1    , 0    , 1    ,
1535      . 0    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1536      . 1    , 1    , 0    , 1    , 0    , 1    , 1    , 1    ,
1537      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1538      . 1    , 1    , 0    , 1    , 0    , 1    , 1    , 1    ,
1539      . 0    , 1    , 0    , 1    , 1    , 1    , 0    , 1    ,
1540      . 0    , 1    , 1    , 1    , 0    , 1    , 0    , 1    , /
1541      data (PenCommand(i),i=11121,11200)/
1542      . 0    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1543      . 1    , 1    , 1    , 0    , 1    , 1    , 1    , 1    ,
1544      . 1    , 1    , 1    , 0    , 1    , 0    , 1    , 1    ,
1545      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1546      . 1    , 0    , 1    , 1    , 1    , 1    , 1    , 0    ,
1547      . 1    , 0    , 1    , 0    , 1    , 1    , 1    , 0    ,
1548      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1549      . 1    , 1    , 1    , 1    , 1    , 0    , 1    , 1    ,
1550      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1551      . 0    , 1    , 1    , 1    , 1    , 1    , 1    , 1    , /
1552      data (PenCommand(i),i=11201,11280)/
1553      . 1    , 1    , 1    , 1    , 0    , 1    , 1    , 1    ,
1554      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 0    ,
1555      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1556      . 1    , 1    , 1    , 0    , 1    , 1    , 1    , 1    ,
1557      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 0    ,
1558      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1559      . 1    , 1    , 1    , 0    , 1    , 0    , 1    , 0    ,
1560      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1561      . 1    , 1    , 1    , 0    , 1    , 0    , 1    , 0    ,
1562      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 1    , /
1563      data (PenCommand(i),i=11281,11360)/
1564      . 1    , 1    , 0    , 1    , 0    , 1    , 0    , 1    ,
1565      . 0    , 1    , 1    , 1    , 0    , 1    , 1    , 1    ,
1566      . 0    , 1    , 1    , 1    , 0    , 1    , 0    , 1    ,
1567      . 0    , 1    , 0    , 1    , 1    , 1    , 0    , 1    ,
1568      . 1    , 1    , 1    , 1    , 1    , 0    , 1    , 1    ,
1569      . 1    , 1    , 1    , 1    , 0    , 1    , 1    , 1    ,
1570      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1571      . 1    , 1    , 1    , 0    , 1    , 0    , 1    , 0    ,
1572      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 0    ,
1573      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 1    , /
1574      data (PenCommand(i),i=11361,11440)/
1575      . 1    , 0    , 1    , 1    , 1    , 1    , 0    , 1    ,
1576      . 1    , 1    , 1    , 1    , 0    , 1    , 1    , 1    ,
1577      . 0    , 1    , 0    , 1    , 1    , 1    , 0    , 1    ,
1578      . 0    , 1    , 0    , 1    , 0    , 1    , 0    , 1    ,
1579      . 0    , 1    , 1    , 1    , 0    , 1    , 1    , 1    ,
1580      . 1    , 1    , 1    , 1    , 1    , 0    , 1    , 1    ,
1581      . 1    , 1    , 1    , 1    , 1    , 0    , 1    , 1    ,
1582      . 1    , 1    , 1    , 1    , 1    , 0    , 1    , 0    ,
1583      . 1    , 1    , 1    , 1    , 1    , 0    , 1    , 0    ,
1584      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 1    , /
1585      data (PenCommand(i),i=11441,11520)/
1586      . 1    , 0    , 1    , 0    , 1    , 0    , 1    , 1    ,
1587      . 1    , 1    , 1    , 1    , 1    , 0    , 1    , 0    ,
1588      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1589      . 1    , 0    , 1    , 1    , 1    , 0    , 1    , 1    ,
1590      . 1    , 1    , 1    , 1    , 1    , 0    , 1    , 1    ,
1591      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 1    ,
1592      . 1    , 1    , 0    , 1    , 1    , 1    , 1    , 1    ,
1593      . 1    , 1    , 1    , 1    , 0    , 1    , 1    , 1    ,
1594      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 0    ,
1595      . 1    , 1    , 1    , 1    , 1    , 1    , 1    , 0    , /
1596      data (PenCommand(i),i=11521,11600)/
```

194

```
1597      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
15 8      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1599      . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 1      ,
1600      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 0      ,
1601      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1602      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1603      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 1      ,
1604      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1605      . 1      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1606      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      /
1607      data (PenCommand(i),i=11601,11680)/
1608      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1609      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1610      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
1611      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1612      . 1      , 1      , 1      , 0      , 1      , 1      , 1      , 0      ,
1613      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1614      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 1      ,
1615      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1616      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1617      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
1618      data (PenCommand(i),i=11681,11760)/
1619      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 0      ,
1620      . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
1621      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1622      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
1623      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1624      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1625      . 1      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1626      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1627      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1628      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
1629      data (PenCommand(i),i=11761,11840)/
1630      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1631      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1632      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1633      . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
1634      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1635      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      ,
1636      . 1      , 0      , 1      , 1      , 1      , 1      , 1      , 0      ,
1637      . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 0      ,
1638      . 1      , 0      , 1      , 1      , 1      , 0      , 1      , 0      ,
1639      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      /
1640      data (PenCommand(i),i=11841,11920)/
1641      . 0      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1642      . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      ,
1643      . 1      , 1      , 1      , 1      , 1      , 0      , 1      , 1      ,
1644      . 1      , 0      , 1      , 0      , 1      , 1      , 1      , 1      ,
1645      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1646      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1647      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1648      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 0      ,
1649      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1650      . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 1      /
1651      data (PenCommand(i),i=11921,12000)/
1652      . 1      , 1      , 0      , 1      , 0      , 1      , 1      , 1      ,
1653      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1654      . 1      , 1      , 1      , 1      , 1      , 1      , 0      , 1      ,
1655      . 0      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1656      . 1      , 1      , 0      , 1      , 1      , 1      , 0      , 1      ,
1657      . 0      , 1      , 0      , 1      , 0      , 1      , 0      , 1      ,
1658      . 0      , 1      , 0      , 1      , 1      , 1      , 1      , 1      ,
1659      . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1660      . 1      , 1      , 1      , 0      , 1      , 0      , 1      , 0      ,
```

```
1661      . 1        , 0        , 1        , 1        , 1        , 1        , 1        , 0        /
1662      data (PenCommand(i),i=12001,12080)/
1663      . 1        , 0        , 1        , 0        , 1        , 0        , 1        , 0        ,
1664      . 1        , 0        , 1        , 0        , 1        , 1        , 1        , 1        ,
1665      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1666      . 1        , 1        , 0        , 1        , 1        , 1        , 0        , 1        ,
1667      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1668      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1669      . 1        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1670      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1671      . 0        , 1        , 1        , 1        , 1        , 0        , 1        , 1        ,
1672      . 1        , 1        , 1        , 1        , 0        , 1        , 0        , 1        /
1673      data (PenCommand(i),i=12081,12160)/
1674      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1675      . 0        , 1        , 0        , 1        , 1        , 1        , 1        , 0        ,
1676      . 1        , 0        , 1        , 0        , 1        , 0        , 1        , 0        ,
1677      . 1        , 0        , 1        , 1        , 1        , 0        , 1        , 0        ,
1678      . 1        , 0        , 1        , 0        , 1        , 0        , 1        , 0        ,
1679      . 1        , 0        , 1        , 0        , 1        , 0        , 1        , 0        ,
1680      . 1        , 0        , 1        , 0        , 1        , 0        , 1        , 0        ,
1681      . 1        , 0        , 1        , 1        , 1        , 1        , 1        , 1        ,
1682      . 1        , 1        , 1        , 1        , 1        , 1        , 1        , 0        ,
1683      . 1        , 1        , 1        , 1        , 1        , 1        , 1        , 1        /
1684      data (PenCommand(i),i=12161,12240)/
1685      . 1        , 1        , 1        , 1        , 1        , 1        , 1        , 1        ,
1686      . 0        , 1        , 1        , 1        , 1        , 1        , 1        , 1        ,
1687      . 1        , 1        , 1        , 1        , 1        , 1        , 1        , 1        ,
1688      . 0        , 1        , 0        , 1        , 0        , 1        , 1        , 1        ,
1689      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1690      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1691      . 0        , 1        , 0        , 1        , 1        , 1        , 1        , 1        ,
1692      . 1        , 1        , 1        , 1        , 1        , 1        , 1        , 1        ,
1693      . 1        , 0        , 1        , 1        , 1        , 1        , 1        , 1        ,
1694      . 1        , 1        , 1        , 1        , 1        , 1        , 1        , 1        /
1695      data (PenCommand(i),i=12241,12320)/
1696      . 0        , 1        , 1        , 1        , 1        , 1        , 1        , 1        ,
1697      . 1        , 1        , 1        , 1        , 1        , 1        , 1        , 1        ,
1698      . 1        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1699      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1700      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1701      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1702      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1703      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1704      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1705      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        /
1706      data (PenCommand(i),i=12321,12400)/
1707      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1708      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1709      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1710      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1711      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1712      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1713      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1714      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1715      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1716      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1
1717      data (PenCommand(i),i=12401,12480)
1718      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1719      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1720      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1721      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1722      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1723      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
1724      . 0        , 1        , 0        , 1        , 0        , 1        , 0        , 1        ,
```

```
1725      . 0      , 1      , 1       , 1       , 1      , 1      , 0      , 1       ,
1726      . 0      , 1      , 0       , 1       , 0      , 1      , 0      , 1       ,
1727      . 0      , 1      , 0       , 1       , 0      , 1      , 0      , 1       /
1728          data (PenCommand(i),i=12481,12560)/
1729      . 0      , 1      , 1       , 1       , 0      , 1      , 0      , 1       ,
1730      . 1      , 1      , 1       , 1       , 1      , 1      , 0      , 1       ,
1731      . 0      , 1      , 0       , 1       , 0      , 1      , 0      , 1       ,
1732      . 0      , 1      , 0       , 1       , 0      , 1      , 1      , 1       ,
1733      . 1      , 1      , 0       , 1       , 1      , 1      , 1      , 1       ,
1734      . 0      , 1      , 0       , 1       , 0      , 1      , 0      , 1       ,
1735      . 1      , 1      , 1       , 1       , 0      , 1      , 0      , 1       ,
1736      . 1      , 1      , 0       , 1       , 0      , 1      , 0      , 1       ,
1737      . 0      , 1      , 0       , 1       , 0      , 1      , 0      , 1       ,
1738      . 0      , 1      , 0       , 1       , 0      , 1      , 0      , 1       /
1739          data (PenCommand(i),i=12561,12640)/
1740      . 1      , 1      , 1       , 1       , 1      , 1      , 1      , 1       ,
1741      . 1      , 1      , 1       , 1       , 1      , 0      , 1      , 0       ,
1742      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1743      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1744      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1745      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1746      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1747      . 1      , 1      , 1       , 1       , 0      , 1      , 0      , 1       ,
1748      . 0      , 1      , 1       , 1       , 0      , 1      , 0      , 1       ,
1749      . 1      , 1      , 0       , 1       , 1      , 1      , 1      , 1       /
1750          data (PenCommand(i),i=12641,12720)/
1751      . 1      , 1      , 1       , 0       , 1      , 0      , 1      , 0       ,
1752      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 1       ,
1753      . 1      , 1      , 1       , 1       , 1      , 0      , 1      , 0       ,
1754      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1755      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1756      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1757      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1758      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1759      . 1      , 0      , 1       , 1       , 1      , 0      , 1      , 0       ,
1760      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       /
1761          data (PenCommand(i),i=12721,12800)/
1762      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1763      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1764      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1765      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1766      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1767      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1768      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1769      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1770      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1771      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       /
1772          data (PenCommand(i),i=12801,12880)/
1773      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1774      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1775      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1776      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1777      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1778      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1779      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1780      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1781      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1782      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       /
1783          data (PenCommand(i),i=12881,12960)/
1784      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1785      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1786      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1787      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
1788      . 1      , 0      , 1       , 0       , 1      , 0      , 1      , 0       ,
```

```
1789         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1790         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1791         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1792         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1793         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      /
1794         data (PenCommand(i),i=12961,1304./
1795         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1796         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1797         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1798         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1799         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1800         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1801         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1802         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1803         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1804         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      /
1805         data (PenCommand(i),i=13041,13120)/
1806         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1807         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1808         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1809         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1810         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1811         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1812         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1813         . 1      , 0      , 1      , 0      , 1      , 0      , 1      , 0      ,
1814         . 1      , 1      , 1      , 1      , 1      , 1      , 1      , 1      ,
1815         . 1      , 1      , 1      , 1      , 0      , 1      , 1      , 1      /
1816
1817         end


0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c-----------------------------------------------------------------------------
0010         subroutine ReadFlightData
0011    c-----------------------------------------------------------------------------
0012    c    read flight (trajectory) data from an input file specified via SFOpen
0013
0014    c.....common block definition files
0015
0016         include 'MapLim.inc'
0017         include 'TicDat.inc'
0018         include 'TrjCom.inc'
0019         include 'TrjLim.inc'
0020
0021    c.....pointer for QuickDraw globals
0022
0023         common   / QDGPtr /         QDG
0024         pointer  / QDGlobals        QDG
0025         integer*4                   jQDGlobals
0026         external                    jQDGlobals
0027
0028    c.....cursor handle
0029
0030         record / CursHandle /       CursorHndl
0031
0032    c.....I/O status flags
0033
```

```
0034          integer*2                ioflag
0035          integer*2                iOpen
0036
0037    c.....character items
0038
0039          character*255            ChrDat
0040          character*4              FilTyp
0041          character*4              fMaker
0042
0043    c-------------------------------------------------------------------------------
0044
0045    c.....get text data file and open it via routine SFOpenFile.  Exit program if
0046    c     unsuccessful.
0047
0048          if ( iOpen.eq.0 ) then
0049             call SFOpenFile ( 8 , iOpen , FilTyp , fMaker )
0050             if ( iOpen.eq.0 ) then
0051                call exit
0052             end if
0053          end if
0054
0055    c.....use watch cursor while reading data
0056
0057          cursorHndl = GetCursor ( %val(4) )
0058          call SetCursor ( %val(cursorHndl.CRHDL^.CRPTR^) )
0059
0060    c.....execute the read loop
0061
0062          ioflag = 0
0063          ntrpts = 0
0064          do while ( ioflag.eq.0 )
0065             ntrpts = ntrpts + 1
0066             read(8,*,iostat=ioflag) TofTab(ntrpts),LngTab(ntrpts),LatTab(ntrpts),
0067        &                           AltTab(ntrpts),JmpTab(ntrpts)
0068             if ( ioflag.ne.0 ) then
0069                ntrpts = ntrpts - 1
0070             end if
0071          end do
0072
0073    c.....close the input file
0074
0075          close ( unit=8 )
0076          iOpen  = 0
0077
0078    c.....determine minimum and maximum limits of data
0079
0080          if ( ntrpts.ne.0 ) then
0081             do i = 1 , ntrpts
0082                if ( i.eq.1 ) then
0083                   MinTof = TofTab (1)
0084                   MinLng = LngTab (1)
0085                   MinLat = LatTab (1)
0086                   MaxTof = TofTab (1)
0087                   MaxLng = LngTab (1)
0088                   MaxLat = LatTab (1)
0089                else
0090                   MinTof = amin1 ( MinTof , TofTab (i) )
0091                   MinLng = amin1 ( MinLng , LngTab (i) )
0092                   minLat = amin1 ( MinLat , LatTab (i) )
0093                   MaxTof = amax1 ( MaxTof , TofTab (i) )
0094                   MaxLng = amax1 ( MaxLng , LngTab (i) )
0095                   MaxLat = amax1 ( MaxLat , LatTab (i) )
0096                end if
0097             end do
```

199

```
0098          end if
0099
0100     c.....determine preliminary plot limits ( not yet enabled )
0101
0102          call AutoScale ( MinTof , MaxTof , ndivmj , tMapMn , tMapMx ,
0103        &                    tDivMj , tDivMi )
0104
0105     c.....reset cursor to arrow
0106
0107          call SetCursor ( %val(QDG^.Arrow) )
0108
0109          return
0110          end
```

```
0001     c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003     !!G toolbox2.finc
0004
0005     c.....Load the ToolBox traps
0006
0007     !!M Inlines.f
0008
0009     c.....Put the following code in the Main segment
0010
0011     !!S Main
0012     c---------------------------------------------------------------------------
Segment Main
0013          subroutine RefreshRunDialog
0014     c---------------------------------------------------------------------------
0015
0016     !!SETC USINGINCLUDES = FALSE
0017          implicit none
0018
0019     c.....common block definition files
0020
0021          include 'Globals.inc'
0022          include 'RunSetup.inc'
0023
0024     c.....temporary storage
0025
0026          record / Rect / refreshTempRect
0027
0028     c---------------------------------------------------------------------------
0029
0030     c.....set to the run dialog port
0031
0032          call SetPort ( %val( GetSelection ) )
0033          ..
0034     c.....save the current contents of tempRect
0035
0036          refreshTempRect = tempRect
0037
0038     c.....draw the Close button
0039
0040          call GetDItem( %val(GetSelection), %val(rCloseButton),
0041        &                    %ref(DType), %ref(DItem), %ref(tempRect) )
0042
0043     c.....draw thick default outline
0044
0045          call PenSize( %val(3), %val(3) )
0046
0047     c.....draw outside the button by one pixel
0048
```

200

```
0049           call InsetRect( %ref(tempRect), %val(-4), %val(-4) )
0050
0051    c.....draw the button outline
0052
0053           call FrameRoundRect( %ref(tempRect), %val(16), %val(16) )
0054
0055    c.....restore the pen size to the default value
0056
0057           call Pensize( %val(1), %val(1) )
0058
0059    c.....draw a line
0060
0061           call MoveTo ( %val(82), %val(65)  )
0062           call LineTo ( %val(82), %val(104) )
0063
0064    c.....restore tempRect
0065
0066           tempRect = refreshTempRect
0067
0068           return
0069           end




0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c-------------------------------------------------------------------------
0010           subroutine ResizeTheMap
0011    c-------------------------------------------------------------------------
0012    c    resize the map window
0013
0014    c.....include common block definition files
0015
0016           include 'FntCom.inc'
0017           include 'MapCom.inc'
0018
0019    c.....item stuff
0020
0021           record / handle /      ItHndl
0022           record / rect /        ItRect
0023           integer*4              ItType
0024           integer*2              ItNmbr
0025           string*255             ItText
0026
0027    c....."resize the map" dialog interface records
0028
0029           common / ResizeMap /   ResizeMapPtr,    iGotResizePtr
0030           record / DialogPtr /   ResizeMapPtr
0031           integer*2              iGotResizePtr
0032
0033    c.....character strings
0034
0035           character*255          ChrDat
0036
0037    c.....dialog interface variables ( note that pointers are i*4 )
0038
0039           integer*4              infront
0040
0041    c.....dialog interface values
```

201

```
0042
0043          data      infront / -1 /
0044
0045   c.....set dialog font to Times ( it is the most compact )
0046
0047          FntNam = 'Times'
0048          call GetFNum ( %val(FntNam) , FntNum )
0049          call setDAfont ( %val(FntNum) )
0050
0051   c.....Get resize map dialog
0052
0053          if ( iGotResizePtr.eq.0 ) then
0054             ResizeMapPtr  = GetNewDialog ( %val(135) , %val(nil) , %val(inFront) )
0055             iGotResizePtr = 1
0056          end if
0057          call SetPort ( %val(ResizeMapPtr) )
0058
0059   c.....bring the dialog window to the front
0060
0061          call ShowWindow ( %val(ResizeMapPtr) )
0062          call SelectWindow ( %val(ResizeMapPtr) )
0063
0064   c.....Highlight the OK button
0065
0066          ItNmbr = 1
0067          call GetDItem ( %val(ResizeMapPtr) , %val(ItNmbr) , %ref(ItType) ,
0068         .                %ref(ItHndl) , %ref(ItRect) )
0069          call PenSize ( %val(3) , %val(3) )
0070          call InsetRect ( %ref(ItRect) , %val(-4) , %val(-4) )
0071          call FrameRoundRect ( %ref(ItRect) , %val(18) , %val(18) )
0072
0073   c.....set and select map window width
0074
0075          ItNmbr = 3                                      .
0076          write(ChrDat,*) MapWidth
0077          ItText = ChrDat
0078          call GetDItem ( %val(ResizeMapPtr) , %val(ItNmbr) , %ref(ItType) ,
0079         .                %ref(ItHndl) , %ref(ItRect) )
0080          call SetIText ( %val(ItHndl) , %val(ItText) )
0081          call SelIText ( %val(ResizeMapPtr) , %val(ItNmbr) , %val(0) , %val(32767) )
0082
0083   c.....set map window height
0084
0085          ItNmbr = 4
0086          write(ChrDat,*) MapHeight
0087          ItText = ChrDat
0088          call GetDItem ( %val(ResizeMapPtr) , %val(ItNmbr) , %ref(ItType) ,
0089         .                %ref(ItHndl) , %ref(ItRect) )
0090          call SetIText ( %val(ItHndl) , %val(ItText) )
0091
0092   c.....loop until either the OK button or the RESET button is clicked.
0093   c     Monitor all other relevant events and update the dialog as necessary.
0094
0095          ItNmbr = 0
0096          do while ( ItNmbr.ne.1 .and. ItNmbr.ne.2 )
0097
0098   c........get number of item hit
0099
0100             call ModalDialog ( %val(nil) , ItNmbr )
0101
0102   c........reset to screen limits
0103
0104             if ( ItNmbr.eq.2 ) then
0105
```

```
0106    c..........width
0107
0108            ItNmbr = 3
0109            write(ChrDat,*) DefWidth
0110            ItText = ChrDat
0111            call GetDItem ( %val(ResizeMapPtr) , %val(ItNmbr) , %ref(ItType) ,
0112        .                   %ref(ItHndl) , %ref(ItRect) )
0113            call SetIText ( %val(ItHndl) , %val(ItText) )
0114            call SelIText ( %val(ResizeMapPtr) , %val(ItNmbr) , %val(0) ,
0115        .                   %val(32767) )
0116
0117    c..........height
0118
0119            ItNmbr = 4
0120            write(ChrDat,*) DefHeight
0121            ItText = ChrDat
0122            call GetDItem ( %val(ResizeMapPtr) , %val(ItNmbr) , %ref(ItType) ,
0123        .                   %ref(ItHndl) , %ref(ItRect) )
0124            call SetIText ( %val(ItHndl) , %val(ItText) )
0125
0126          end if
0127
0128        end do
0129
0130    c.....get map window width
0131
0132        ItNmbr = 3
0133        call GetDitem ( %val(ResizeMapPtr) , %val(ItNmbr) , %ref(ItType) ,
0134        .               %ref(ItHndl) , %ref(ItRect) )
0135        call GetIText ( %val(ItHndl) , %val(ItText) )
0136        ChrDat = ItText
0137        if ( ChrDat.ne.' ' ) then
0138           read(ChrDat,*,iostat=ioflag) tmp1
0139           if ( ioflag.ne.0 ) then
0140              tmp1   = 0.0
0141           end if
0142        else
0143           tmp1   = 0.0
0144        end if
0145        MapWidth = tmp1
0146
0147    c.....get map window height
0148
0149        ItNmbr = 4
0150        call GetDitem ( %val(ResizeMapPtr) , %val(ItNmbr) , %ref(ItType) ,
0151        .               %ref(ItHndl) , %ref(ItRect) )
0152        call GetIText ( %val(ItHndl) , %val(ItText) )
0153        ChrDat = ItText
0154        if ( ChrDat.ne.' ' ) then
0155           read(ChrDat,*,iostat=ioflag) tmp1
0156           if ( ioflag.ne.0 ) then
0157              tmp1   = 0.0
0158           end if
0159        else
0160           tmp1   = 0.0
0161        end if
0162        MapHeight = tmp1
0163
0164    c.....hide dialog
0165
0166        call HideWindow ( %val(ResizeMapPtr) )
0167
0168        return
0169        end
```

203

```
0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c------------------------------------------------------------------------
0010          subroutine RunSetupDialog
0011    c------------------------------------------------------------------------
0012
0013    !!SETC USINGINCLUDES = FALSE
0014          implicit none
0015
0016    c.....common block definition files
0017
0018          include 'FileInfo.inc'
0019          include 'FileMenu.inc'
0020          include 'Globals.inc'
0021          include 'MapMenu.inc'
0022          include 'RunSetup.inc'
0023          include 'Traj.inc'
0024          include 'TrjCom.inc'
0025
0026    c.....set up pointer for QuickDraw globals
0027
0028          pointer / QDGlobals /      qdg
0029          common  / QDGPtr    /      qdg
0030
0031    c.....external function declarations
0032
0033          external  RunSetupFilter
0034          logical*1 RunSetupFilter
0035
0036          external  CollectRunInput
0037          integer*2 CollectRunInput
0038
0039    c.....integers for error handling
0040
0041          integer*2 ErrorItem
0042          integer*2 iopen
0043          integer*4 ioserr
0044
0045    c.....integer for popup menu selection
0046
0047          integer*2 theSelection
0048
0049    c.....boolean for dialog handle loop
0050
0051          logical*1 ExitDialog
0052
0053    c.....intermediate text strings
0054
0055          string*255 ItemText
0056          string*255 Filename
0057
0058    c.....character intermediate
0059
0060          character*255 CharData
0061
0062    c.....other
```

```
0063
0064          character*4 FilTyp
0065          character*4 fmaker
0066
0067    c.....cursor handle
0068
0069          record / CursHandle /        CursorHndl
0070
0071    c.....STR handling
0072
0073          record / handle        /      RezHndl
0074          record / StringHandle /      ST.'ndl
0075
0076          string*255              tempString
0077
0078    c.....pop-up menu handling
0079
0080          record / MenuHandle /    PopupHndl
0081
0082    c.....dialog template handling
0083
0084          record / DialogTHndl    / DlogHndl
0085          record / DialogTemplate / DlogTemp
0086          record / Rect           / RunRect
0087
0088    c.....screen position info
0089
0090          integer*2               menuHeight
0091          integer*4               left, bottom, top, right
0092          integer*4               dialogHeight, dialogWidth
0093
0094    c.....dimensions of runsetup dialog box
0095
0096          parameter          ( dialogHeight      =  302 )
0097          parameter          ( dialogWidth       =  491 )
0098
0099    c-------------------------------------------------------------------------------
0100
0101    c.....get the previous grafPort
0102
0103          call GetPort ( %val( SavedPort ))
0104
0105    c.....get the handle and the template for the dialog resource
0106
0107          RezHndl  = GetResource( %val('DLOG'), %val(rRunSetupDLOG) )
0108          call HNoPurge( %val(RezHndl) )
0109          DlogHndl = RezHndl
0110          call HNoPurge( %val(DlogHndl) )
0111
0112    c.....get the menuHeight (don't assume it is fixed at 20!)
0113
0114          menuHeight = GetMBarHeight()
0115
0116    c.....get the screen extents (use i*4 for screen math per Mac Tech Note 117)
0117
0118          left   = qdg^.screenBits.bounds.left
0119          right  = qdg^.screenBits.bounds.right
0120          bottom = qdg^.screenBits.bounds.bottom
0121          top    = qdg^.screenBits.bounds.top + int4(menuHeight)
0122
0123    c.....set the four corners of the dialog
0124
0125          RunRect.left   = int2( ( (right - left) - dialogWidth  ) / 2 )
0126          RunRect.top    = int2(( ( (bottom - top) - dialogHeight ) / 2 ) + menuHeight)
```

```
0127            RunRect.bottom = RunRect.top  + int2(dialogHeight)
0128            RunRect.right  = RunRect.left + int2(dialogWidth)
0129
0130            DlogHndl.DtH^.DtP^.boundsRect = RunRect
0131            call ChangedResource( *val(RezHndl) )
0132            call WriteResource( *val(RezHndl) )
0133            call HPurge( *val(RezHndl) )
0134            call HPurge( *val(DlogHndl) )
0135
0136    c.....bring in the dialog resources
0137
0138            GetSelection = GetNewDialog( *val(rRunSetupDLOG),
0139          &                             *val(nil), *val(inFront) )
0140            call ShowWindow  ( *val(GetSelection) )
0141            call SelectWindow( *val(GetSelection) )
0142            call SetPort     ( *val(GetSelection) )
0143
0144            TheDialogPtr.DPk = GetSelection.DP
0145            ThisEditText.TEH = TheDialogPtr.DPk
0146            call HLock( *val(ThisEditText))
0147
0148    c.....set the TE point size
0149            ThisEditText.TEH^.TEP^.txSize = int2(12)
0150
0151    c.....set the window point size with QuickDraw procedure
0152            call TextSize( *val(int2(12)) )
0153
0154    c.....set the TE font ID
0155            ThisEditText.TEH^.TEP^.txFont = systemFont
0156
0157    c.....set the window font ID with QuickDraw procedure
0158            call TextFont( *val(systemFont) )
0159
0160    c.....set the TE font ascent, descent and leading values
0161            ThisEditText.TEH^.TEP^.fontAscent = int2(12)
0162            ThisEditText.TEH^.TEP^.lineHeight = int2(12 + 3 + 1)
0163
0164            call HUnlock( *val(ThisEditText) )
0165
0166    c.....set up initial conditions for dialog items
0167
0168            rDegreeSelection   = rDegWestButton
0169            rDistanceSelection = rMeters
0170            rTimeSelection     = rSeconds
0171            AscentSelection    = 'Ascent Profile'
0172            ClimateSelection   = 1
0173
0174    c.....set up degrees-west button
0175
0176            call GetDItem( *val(GetSelection), *val(rDegWestButton),
0177          &               *ref(DType), *ref(DItem), *ref(tempRect) )
0178            CItem.CtlH = DItem.bhdl
0179            call SetCtlValue( *val(CItem), *val(int2(1) ) )
0180            call SetupTheItem( GetSelection, rDegWestButton,
0181          &                    .true., .true., .true.. .false., tempRect
0182
0183    c.....set up degrees-east button
0184
0185            call GetDItem( *val(GetSelection), *val(rDegEastButton),
0186          &               *ref(DType), *ref(CItem), *ref(tempRect)
0187            CItem.CtlH = CItem.bhdl
0188            call SetCtlValue( *val(CItem), *val(int2(0) )
0189            call SetupTheItem( GetSelection, rDegEastButton,
0190          &                    .true., .true., .true., .false.. tempRect. 0.
```

```
0191
0192      c.....set up meters button
0193
0194          call GetDItem( %val(GetSelection), %val(rMeters),
0195      &                  %ref(DType), %ref(DItem), %ref(tempRect) )
0196          CItem.CtlH = DItem.bhdl
0197          call SetCtlValue( %val(CItem), %val( int2(1) ) )
0198          call SetupTheItem( GetSelection, rMeters,
0199      &                          .true., .true., .true., .false., tempRect, 0, 0 )
0200
0201      c.....set up kilometers button
0202
0203          call GetDItem( %val(GetSelection), %val(rKilometers),
0204      &                  %ref(DType), %ref(DItem), %ref(tempRect) )
0205          CItem.CtlH = DItem.bhdl
0206          call SetCtlValue( %val(CItem), %val( int2(0) ) )
0207          call SetupTheItem( GetSelection, rKilometers,
0208      &                          .true., .true., .true., .false., tempRect, 0, 0 )
0209
0210      c.....set up seconds button
0211
0212          call GetDItem( %val(GetSelection), %val(rSeconds),
0213      &                  %ref(DType), %ref(DItem), %ref(tempRect) )
0214          CItem.CtlH = DItem.bhdl
0215          call SetCtlValue( %val(CItem), %val(int2(1) ) )
0216          call SetupTheItem( GetSelection, rSeconds,
0217      &                          .true., .true., .true., .false., tempRect, 0, 0 )
0218
0219      c.....set up minutes button
0220
0221          call GetDItem( %val(GetSelection), %val(rMinutes),
0222      &                  %ref(DType), %ref(DItem), %ref(tempRect) )
0223          CItem.CtlH = DItem.bhdl
0224          call SetCtlValue( %val(CItem), %val(int2(0) ) )
0225          call SetupTheItem( GetSelection, rMinutes,
0226      &                          .true., .true., .true., .false., tempRect, 0, 0 )
0227
0228      c.....set up hours button
0229
0230          call GetDItem( %val(GetSelection), %val(rHours),
0231      &                  %ref(DType), %ref(DItem), %ref(tempRect) )
0232          CItem.CtlH = DItem.bhdl
0233          call SetCtlValue( %val(CItem), %val(int2(0) ) )
0234          call SetupTheItem( GetSelection, rHours,
0235      &                          .true., .true., .true., .false., tempRect, 0, 0 )
0236
0237      c.....set up wind model selector
0238
0239          call GetDItem( %val(GetSelection), %val(rWindModelSelector),
0240      &                  %ref(DType), %ref(DItem), %ref(tempRect) )
0241          CItem.CtlH = DItem.bhdl
0242          call SetCtlMax( %val(CItem), %val(rWindModelSelectorPopup) )
0243          call SetCtlValue( %val(CItem), %val(ClimateSelection) )
0244          call SetupTheItem( GetSelection, rWindModelSelector,
0245      &                          .true., .true., .true., .false., tempRect, 0, 0 )
0246
0247      c.....set up ascent selector
0248
0249          call GetDItem( %val(GetSelection), %val(rAscentSelector),
0250      &                  %ref(DType), %ref(DItem), %ref(tempRect) )
0251          CItem.CtlH = DItem.bhdl
0252          call SetCtlMax( %val(CItem), %val(rAscentSelectorPopup) )
0253          call SetCtlValue( %val(CItem), %val(int2(1) ) )
0254          call SetupTheItem( GetSelection, rAscentSelector,
```

207

```
0255          &                            .true., .true., .true., .false., tempRect, 0, 0 )
0256
0257      c.....check to see if an old file's data is available
0258
0259          if ( iGotOldFile ) then
0260
0261      c.......get the old value of latitude
0262
0263              call GetDItem( %val(GetSelection), %val(rLatitude),
0264          &                    %ref(DType), %ref(DItem), %ref(tempRect) )
0265              RezHndl = GetResource( %val('STR '), %val(rOldLatitude) )
0266              StrHndl.shdl = RezHndl.bhdl
0267              tempString = StrHndl.shdl^.sptr^
0268              call SetIText( %val(DItem), %val(tempString) )
0269
0270      c.......get the old value of longitude
0271
0272              call GetDItem( %val(GetSelection), %val(rLongitude),
0273          &                    %ref(DType), %ref(DItem), %ref(tempRect) )
0274              RezHndl = GetResource( %val('STR '), %val(rOldLongitude) )
0275              StrHndl.shdl = RezHndl.bhdl
0276              tempString = StrHndl.shdl^.sptr^
0277              call SetIText( %val(DItem), %val(tempString) )
0278
0279      c.......get the old value of flight duration
0280
0281              call GetDItem( %val(GetSelection), %val(rDuration),
0282          &                    %ref(DType), %ref(DItem), %ref(tempRect) )
0283              RezHrdl = GetResource( %val('STR '), %val(rOldDuration) )
0284              StrHndl.shdl = RezHndl.bhdl
0285              tempString = StrHndl.shdl^.sptr^
0286              call SetIText( %val(DItem), %val(tempString) )
0287
0288      c.......get the old value of altitude
0289
0290              call GetDItem( %val(GetSelection), %val(rAltitude),
0291          &                    %ref(DType), %ref(DItem), %ref(tempRect) )
0292              RezHndl = GetResource( %val('STR '), %val(rOldAltitude) )
0293              StrHndl.shdl = RezHndl.bhdl
0294              tempString = StrHndl.shdl^.sptr^
0295              call SetIText( %val(DItem), %val(tempString) )
0296
0297      c.......get the old value of Mission text
0298
0299              call GetDItem( %val(GetSelection), %val(rMissionLabel),
0300          &                    %ref(DType), %ref(DItem), %ref(tempRect) )
0301              RezHndl = GetResource( %val('STR '), %val(rOldMissionText) )
0302              StrHndl.shdl = RezHndl.bhdl
0303              tempString = StrHndl.shdl^.sptr^
0304              call SetIText( %val(DItem), %val(tempString) )
0305
0306      c.......get the old value of wind model selection
0307
0308              call GetDItem( %val(GetSelection), %val(rWindModelSelector),
0309          &                    %ref(DType , %ref(DItem), %ref(tempRect) )
0310              CItem.CtlH = DItem.bhdl
0311              StrHndl    = GetString( %val(rOldClimate) )
0312              tempString = StrHndl.shdl^.sptr^
0313              CharData   = tempString
0314              read(CharData,*) ClimateSelection
0315              call SetCtlValue( %val(CItem), %val(ClimateSelection) )
0316
0317      c.......get the old value of ascent profile selection
0318
```

208

```
0319             StrHndl        = GetString( %val(rOldAscent) )
0320             tempString     = StrHndl.shdl^.sptr^
0321             CharData       = tempString
0322             PopupHndl.menuH = GetMHandle( %val(rAscentSelectorPopup) )
0323             call SetItem( %val(PopupHndl), %val(int2(1)), %val(tempString) )
0324
0325     c.......get the old value of the deg West/deg East radio button
0326
0327             call ClearDegreeGroup
0328             StrHndl    = GetString ( %val(rOldDegRadio) )
0329             tempString = StrHndl.shdl^.sptr^
0330             CharData   = tempString
0331             read(CharData,*) rDegreeSelection
0332             call GetDItem( %val(GetSelection), %val(rDegreeSelection),
0333          &                 %ref(DType), %ref(DItem), %ref(tempRect) )
0334             CItem.CtlH = DItem.bhdl
0335             call SetCtlValue( %val(CItem), %val(int2(1) ) )
0336
0337     c.......get the old value of the m/km radio button
0338
0339             call ClearDistanceGroup
0340             StrHndl    = GetString ( %val(rOldDistRadio) )
0341             tempString = StrHndl.shdl^.sptr^
0342             CharData   = tempString
0343             read(CharData,*) rDistanceSelection
0344             call GetDItem( %val(GetSelection), %val(rDistanceSelection),
0345          &                 %ref(DType), %ref(DItem), %ref(tempRect) )
0346             CItem.CtlH = DItem.bhdl
0347             call SetCtlValue( %val(CItem), %val(int2(1) ) )
0348
0349     c.......get the old value of the sec/min/hrs radio button
0350
0351             call ClearTimeGroup
0352             StrHndl    = GetString ( %val(rOldTimeRadio) )
0353             tempString = StrHndl.shdl^.sptr^
0354             CharData   = tempString
0355             read(CharData,*) rTimeSelection
0356             call GetDItem( %val(GetSelection), %val(rTimeSelection),
0357          &                 %ref(DType), %ref(DItem), %ref(tempRect) )
0358             CItem.CtlH = DItem.bhdl
0359             call SetCtlValue( %val(CItem), %val(int2(1) ) )
0360
0361     c.......close the resource file for now
0362
0363             call CloseResFile( %val(RefNum) )
0364
0365          endif
0366
0367     c.....call routine to draw any lists, lines, or rectangles
0368
0369          call RefreshRunDialog
0370
0371     c.....do not exit the dialog handle loop yet
0372
0373          ExitDialog = .false.
0374
0375     c.....********************************
0376     c.....** start of dialog handle loop **
0377     c.....********************************
0378
0379          do while (.not.ExitDialog)
0380
0381     c.......get number of item hit
0382
```

```
0383                call ModalDialog ( RunSetupFilter , ItemHit )
0384
0385     c.......check for update
0386
0387            if( ItemHit .eq. 32000 ) then
0388              call RefreshRunDialog
0389              call EndUpdate( %val(GetSelection) )
0390            else
0391              call GetDItem( %val(GetSelection), %val(ItemHit),
0392        &                   %ref(DType), %ref(DItem), %ref(tempRect) )
0393              CItem.CtlH = DItem.bhdl
0394            endif
0395
0396     c.......check for Close button
0397
0398            if( ItemHit .eq. rCloseButton ) then
0399     c.........first enable the New and Open Mission buttons
0400              call MenuSet( FileMenuID, FileItemNewMission,  .true. )
0401              call MenuSet( FileMenuID, FileItemOpenMission, .true. )
0402              ExitDialog = .true.
0403            endif
0404
0405     c.......check for Run button
0406
0407            if( ItemHit .eq. rRunButton ) then
0408              ErrorItem = CollectRunInput()
0409              if ( ErrorItem.ne.0 ) then
0410                call SysBeep( %val(int2(20) ) )
0411                call SeliText( %val(GetSelection), %val(ErrorItem),
0412        &                     %val(int2(0)), %val(int2(32767)) )
0413              else
0414     c...........use watch cursor while running
0415                cursorHndl = GetCursor ( %val(int2(4)) )
0416                call SetCursor ( %val(cursorHndl.CRHDL^.CRPTR^) )
0417                call RunTraj( AscentSelection, int4(ClimateSelection),
0418        &                   xDuration, xLatitude, xLongitude, xAltitude)
0419     c...........reset cursor to arrow
0420                call SetCursor ( %val(QDG^.Arrow) )
0421                call SysBeep( %val(int2(20) ) )
0422                call SysBeep( %val(int2(20) ) )
0423
0424              endif
0425            endif
0426
0427     c.......check for Map button
0428
0429            if( ItemHit .eq. rMapButton ) then
0430     c.........first enable the map menu items
0431              call MenuSet( MapMenuID, itemGetNewDataSet, .true. )
0432              call MenuSet( MapMenuID, itemNewMap,        .true. )
0433              call MenuSet( MapMenuID, itemSaveMap,       .true. )
0434              call MenuSet( MapMenuID, itemDone,          .true. )
0435
0436              call MapIt
0437
0438     c.........now disable the entire map menu
0439              call MenuSet( MapMenuID, 0, .false. )
0440
0441            endif
0442
0443     c.......check for Save button
0444
0445            if( ItemHit .eq. rSaveButton ) then
0446              ErrorItem = CollectRunInput()
```

210

```
0447                    if ( ErrorItem.ne.0 ) then
0448                      call SysBeep( %val(int2(20) ) )
0449                      call SeliText( %val(GetSelection), %val(ErrorItem),
0450          &                          %val(0), %val(32767) )
0451                    else
0452                      call SaveMissionFile
0453                    endif
0454                  endif
0455
0456    c.......check for degrees-west radio button
0457
0458            if( ItemHit .eq. rDegWestButton ) then
0459              call ClearDegreeGroup
0460              call SetCtlValue( %val(CItem), %val(int2(1) ) )
0461              rDegreeSelection = rDegWestButton
0462            endif
0463
0464    c.......check for degrees-east radio button
0465
0466            if( ItemHit .eq. rDegEastButton ) then
0467              call ClearDegreeGroup
0468              call SetCtlValue( %val(CItem), %val(int2(1) ) )
0469              rDegreeSelection = rDegEastButton
0470            endif
0471
0472    c.......check for meters radio button
0473
0474            if( ItemHit .eq. rMeters ) then
0475              call ClearDistanceGroup
0476              call SetCtlValue( %val(CItem), %val(int2(1) ) )
0477              rDistanceSelection = rMeters
0478            endif
0479
0480    c.......check for kilometers radio button
0481
0482            if( ItemHit .eq. rKilometers ) then
0483              call ClearDistanceGroup
0484              call SetCtlValue( %val(CItem), %val(int2(1) ) )
0485              rDistanceSelection = rKilometers
0486            endif
0487
0488    c.......check for seconds radio button
0489
0490            if( ItemHit .eq. rSeconds ) then
0491              call ClearTimeGroup
0492              call SetCtlValue( %val(CItem), %val(int2(1) ) )
0493              rTimeSelection = rSeconds
0494            endif
0495
0496    c.......check for minutes radio button
0497
0498            if( ItemHit .eq. rMinutes ) then
0499              call ClearTimeGroup
0500              call SetCtlValue( %val(CItem), %val(int2(1) ) )
0501              rTimeSelection = rMinutes
0502            endif
0503
0504    c.......check for hours radio button
0505
0506            if( ItemHit .eq. rHours ) then
0507              call ClearTimeGroup
0508              call SetCtlValue( %val(CItem), %val(int2(1) ) )
0509              rTimeSelection = rHours
0510            endif
```

```
0511
0512    c.......check for wind model selection
0513
0514             if( ItemHit .eq. rWindModelSelector ) then
0515                theSelection = GetCtlValue( %val(CItem) )
0516                select case(theSelection)
0517                  case(1)
0518                    ClimateSelection = 1
0519                  case(2)
0520                    ClimateSelection = 2
0521                  case(3)
0522                    ClimateSelection = 3
0523                  case(4)
0524                    ClimateSelection = 4
0525                  case(5)
0526                    ClimateSelection = 5
0527                  case(6)
0528                    ClimateSelection = 6
0529                  case(7)
0530                    ClimateSelection = 7
0531                  case(8)
0532                    ClimateSelection = 8
0533                  case(9)
0534                    ClimateSelection = 9
0535                  case(10)
0536                    ClimateSelection = 10
0537                  case(11)
0538                    ClimateSelection = 11
0539                  case(12)
0540                    ClimateSelection = 12
0541                  case(13)
0542                    ClimateSelection = 13
0543                  case default
0544                    ClimateSelection = 1
0545                end select
0546             endif
0547
0548    c.......check for ascent profile selection
0549
0550             if( ItemHit .eq. rAscentSelector ) then
0551                theSelection = GetCtlValue( %val(CItem) )
0552
0553                select case(theSelection)
0554
0555                case(1)
0556    c.............find what the menu shows and use that name
0557                    PopupHndl.menuH = GetMHandle( %val(rAscentSelectorPopup) )
0558                    call GetItem( %val(PopupHndl), %val(int2(1)), %ref(tempString))
0559                    AscentSelection = tempString
0560
0561                case(2)
0562    c.............get the name of the user's file; stick it in the popup menu
0563                    call SFOpenAscentFile( iopen, Filename )
0564                    if( iopen.eq.1 ) then
0565    c...............change the menu item
0566                        PopupHndl.menuH = GetMHandle( %val(rAscentSelectorPopup) )
0567                        call SetItem( %val(PopupHndl), %val(int2(1)), %val(Filename) )
0568                        AscentSelection = Filename
0569                    endif
0570                end select
0571             endif
0572
0573    c.....******************************
0574    c.....** end of dialog handle loop **
```

```
0575    c......*******************************
0576
0577         end do
0578
0579    c.....restore the port
0580
0581         call SetPort( %val(SavedPort) )
0582
0583         call DisposDialog( %val(GetSelection) )
0584
0585         return
0586         end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Main segment
0010
0011    !!S Main
0012    c-------------------------------------------------------------------------
Segment Main
0013         logical*1 function RunSetupFilter ( %val(theDialog), theEvent, ItemHit )
0014    c-------------------------------------------------------------------------
0015    c    process the relevant events which may occur while the Run Setup dialog
0016    c    is in use.
0017
0018    !!SETC USINGINCLUDES = FALSE
0019         implicit none
0020
0021    c.....common block definition files
0022
0023         include 'RunSetup.inc'
0024
0025    c.....other declarations
0026
0027         record / DialogPtr   / theDialog
0028         record / EventRecord / theEvent
0029         record / WindowPtr   / WinPtr
0030         record / Point       / MyPt
0031
0032    c.....integers
0033
0034         integer*2 WindowPart
0035         integer*2 ChCode
0036         integer*2 CmdCode
0037
0038    c.....character for key handling
0039
0040         character*1 Ch
0041
0042    c-------------------------------------------------------------------------
0043
0044    c.....default: Modal Dialog should process the event
0045
0046         RunSetupFilter = .false.
0047
0048    c.....handle an update event
0049
```

213

```
0050            if (theEvent.what .eq. updateEvt) then
0051                WindowPart = FindWindow ( %val(TheEvent.where) , %ref(WinPtr) )
0052
0053    c.......if in the dialog then only do an update
0054
0055            if ( WinPtr.WP .eq. theDialog.DP ) then
0056               call BeginUpdate( %val(theDialog) )
0057               call DrawDialog( %val(theDialog) )
0058               RunSetupFilter = .true.
0059               ItemHit = 32000
0060            endif
0061
0062    c.....handle a mouse event
0063
0064            else if ( theEvent.what .eq. mouseDown ) then
0065               MyPt = theEvent.where
0066               call GlobalToLocal ( %ref(MyPt) )
0067
0068    c.....handle a keydown event
0069
0070            else if ( theEvent.what .eq. keyDown ) then
0071               ChCode = jiand ( TheEvent.message , CharCodeMask )
0072               Ch      = char ( ChCode )
0073
0074    c.......check for the Cmd key depression
0075               CmdCode = jiand ( theEvent.modifiers , CmdKey )
0076
0077               if ( CmdCode.ne.0 ) then
0078                  if ( Ch.eq.'x' .or. Ch.eq.'X' ) then
0079                     call DlgCut( %val(theDialog) )
0080                     RunSetupFilter = .true.
0081                  else if ( Ch.eq.'c' .or. Ch.eq.'C' ) then
0082                     call DlgCopy( %val(theDialog) )
0083                     RunSetupFilter = .true.
0084                  else if ( Ch.eq.'v' .or. Ch.eq.'V' ) then
0085                     call DlgPaste( %val(theDialog) )
0086                     RunSetupFilter = .true.
0087                  endif
0088               endif
0089
0090    c.......return ItemHit=1 for <Return> or <Enter>
0091               if ( ChCode.eq.3 .or. ChCode.eq.13 ) then
0092                  ItemHit         = 1
0093                  RunSetupFilter = .true.
0094               end if
0095
0096            endif
0097
0098            return
0099            end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G ToolBox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009            SUBROUTINE RUNTRAJ (INFILE,CLIMATE,FLIGHT_TIME,INIT_LAT,
0010       &                        INIT_LON,  INITIAL_ALT)
0011    C
0012            EXTERNAL FUNCNT
```

```
0013   C
0014          LOGICAL FIRST, OFF_EAST, OFF_WEST
0015          DIMENSION PARMIN(10),XMEAN(10),STDDEV(10)
0016          DIMENSION CURR_POSITION(3),INIT_POSITION(3)
0017   C
0018          REAL*4 SLOPE,Y_INTERCEPT,TOTAL_SEG_DIST,SEGMENT_DIST
0019          REAL*4 INIT_SEG_VEL,FIN_SEG_VEL,FUNCNT,PERM_INT_STEP
0020          REAL*4 ANGLE_RESOLUTION,DIFFTM,INIT_SEG_TIME,DIFFGT
0021          REAL*4 FLIGHT_TIME,INITIAL_ALT,INTEGRAT_STEP
0022          REAL*4 FIN_SEG_TIME,FIN_INTEGRAT_LMT
0023          REAL*4 DIFLAT,DIFLON,TOLERANCE,INIT_INTEGRAT_LMT
0024          REAL*4 WIND_AZIMUTH,WIND_VEL,INIT_POSITION,CURR_POSITION
0025          REAL*4 AE,BE,RADIUS_ERTH,T2TOF,DELTAT,ERTH_ROTATION,R2D,DIFFNC
0026          REAL*4 XLATSNG,XLNGSNG,GRD_RANGE,UGH,VGH
0027          REAL*4 LON_ERTH_ROT,INIT_EAST_WIND,INIT_NORTH_WIND,TOTAL_EAST
0028          REAL*4 DIST_EAST,DIST_NORTH,FINAL_ALT,TOTAL_NORTH
0029          REAL*4 NEW_LAT,NEW_LON,INIT_LAT,INIT_LON,LAT_DEG,LON_DEG
0030          REAL*8 FSIM
0031   C
0032          INTEGER*4 ICHECK,IS,CLIMATE
0033          INTEGER*4 PERCNT_DONE
0034   C
0035          LOGICAL*1 SegmentsFinished
0036   C
0037          CHARACTER INFILE*255, TAB*1
0038   C
0039          DATA TAB /9/
0040          DATA R2D /57.29578/
0041          DATA ERTH_ROTATION  /7.2921151E-05/
0042          DATA IS /0/
0043
0044          COMMON /DRVOUT/ FSIM(13),DFSIM(13)
0045          COMMON /DRVOT2/ ALP    ,ALPDOT,BTA    ,BTADOT,HSIM   ,HDOT   ,
0046          .                    V      ,VDOT  ,X       ,XDOT  ,Y       ,YDOT  ,VI
0047          EQUIVALENCE (FSIM( 1),T       ),(FSIM(12),LAT_RAD)
0048          EQUIVALENCE (FSIM(13),LON_RAD)
0049   C
0050          COMMON /PLTOUT/ HWLS   ,HWLL   ,VDS    ,VTS    ,VUS    ,VDL    ,VTL    ,
0051          &                      VUL    ,RDS    ,RTS    ,RVS    ,RDL    ,RTL    ,RVL    ,
0052          &                      RNDL   ,RNDS   ,RNTL   ,RNTS   ,RNUL   ,RNUS   ,RNVL   ,
0053          &                      RNVS   ,RNDLM  ,RNDSM  ,RNTLM  ,RNTSM  ,RNULM  ,RNUSM  ,
0054          &                      RNVLM  ,RNVSM
0055   C
0056   C.... GRAM PROGRAM COMMON BLOCKS.
0057   C
0058          INCLUDE 'naspgcom.inc'
0059   C
0060   C.... DRYDEN SIM COMMON BLOCKS.
0061   C
0062          INCLUDE 'naspcom.inc'
0063   C
0064          include 'traj.inc'
0065          include 'trjcom.inc'
0066          include 'globals.inc'
0067
0068          record / GrafPort      /  SavedPort
0069          record / rect   /         ItRect
0070          integer*4                 ItType
0071          string*255                ItText
0072          string*255                PrText
0073          character*254             Char_Data
0074          record / DialogPtr /      StatPtr
0075          record / handle /         ItHndl
0076          integer*4                 last_value
```

215

```
0077    C
0078    C--------------------------------------------------------------------------
0079    C
0080           RADIUS_ERTH = 6375416.785
0081    C
0082           CALL OPENIT(INIT_LAT,INIT_LON,INFILE,FLIGHT_TIME,CLIMATE)
0083    C
0084           last_value = 0
0085
0086    c.....get the previous grafPort
0087
0088           call GetPort ( %val( SavedPort ))
0089
0090    c.....display the status dialog
0091
0092           StatPtr = GetNewDialog ( %val(int2(129)) , %val(nil) , %val(inFront) )
0093           call ShowWindow  ( %val(StatPtr) )
0094             call SelectWindow( %val(StatPtr) )
0095           call SetPort     ( %val(StatPtr) )
0096
0097           call DrawDialog( %val(StatPtr) )
0098
0099    c.....get the handle for the percent complete text box (item #3)
0100
0101           PrText = '0 %'
0102           call GetDItem ( %val(StatPtr) , %val(int2(3)) , %ref(ItType) , %ref(ItHndl) ,
0103          .               %ref(ItRect) )
0104           call SetIText ( %val(ItHndl) , %val(PrText) )
0105    c
0106    C.... PERFORM INITIALIZATION.
0107    C
0108           FIRST    = .TRUE.
0109           OFF_EAST = .FALSE.
0110           OFF_WEST = .FALSE.
0111           TOTAL_SEG_DIST   = 0.0
0112           FIN_INTEGRAT_LMT = 0.0
0113           TOTAL_EAST  = 0.0
0114           TOTAL_NORTH = 0.0
0115           DELTAT = 0.0
0116           DIFFGT = 0.0
0117           DIFFNC = 0.0
0118           DIFFTM = 0.0
0119    C
0120    C      SUBROUTINE GRAMIN PROVIDES THE BALLOON DRIFT SIMULATION WITH
0121    C      INITIAL VALUES FOR THE LATITUDE (INIT_LAT (DEG)), LONGITUDE
0122    C      (INIT_LON (DEG)), ALTITUDE (H1 (KM)), AND THE RADIUS OF THE EARTH
0123    C      (RADIUS_ERTH (M)).
0124    C
0125           IF(CLIMATE .NE. 13)THEN
0126             CALL GRAMIN (INIT_LAT, INIT_LON, INITIAL_ALT, CLIMATE)
0127           ENDIF
0128    C
0129    C      INITIAL INPUTS FOR RTGRAM AND THE SUBROUTINES.
0130    C
0131           LAT_RAD = INIT_LAT/R2D
0132           LON_RAD = INIT_LON/R2D
0133           XLATSNG = SNGL(LAT_RAD)
0134           XLNGSNG = SNGL(LON_RAD)
0135           INIT_INTEGRAT_LMT = 0.0
0136           TOTAL_EAST  = 0.0
0137           TOTAL_NORTH = 0.0
0138           TOLERANCE   = 0.0001
0139           ANGLE_RESOLUTION = 1.0
0140           NO_OF_PTS = 1
```

216

```
0141          AE  = 6378134.999
0142          BE  = 6356750.499
0143    C
0144    C     NOGAPS/GRAMRT DERIVES THE WIND COMPONENTS, EAST (UGH (M/S))
0145    C     AND NORTH (VGH (M/S)).  THESE COMPONENTS ARE BASED UPON THE
0146    C     INITIAL LATITUDE (INIT_LAT (DEG)/LAT_RAD (RAD)), LONGITUDE
0147    C     (INIT_LON (DEG)/LON_RAD (RAD)), AND ALTITUDE (INTALT
0148    C     (M)/HSIM (FEET)).
0149    C
0150          IF(CLIMATE .EQ. 13)THEN
0151            CALL NOGAPS(INITIAL_ALT, INIT_LAT, INIT_LON, UGH, VGH, FIRST)
0152          ELSE
0153            HSIM = INITIAL_ALT * 3.28084
0154            CALL GRAMRT (FIRST)
0155          ENDIF
0156    C
0157          IF (VGH.EQ.0.0) THEN
0158            WIND_AZIMUTH = 90.0
0159          ELSE
0160            WIND_AZIMUTH = (ATAN(UGH/VGH))*R2D
0161          ENDIF
0162          IF( VGH.LT.0.0) THEN
0163            WIND_AZIMUTH = 180.0+WIND_AZIMUTH
0164          ELSEIF( UGH.LT.0.0 ) THEN
0165            WIND_AZIMUTH = 360.0+WIND_AZIMUTH
0166          ENDIF
0167    C
0168          WIND_VEL = SQRT((UGH**2) + (VGH**2))
0169    C
0170          INIT_EAST_WIND = UGH
0171          INIT_NORTH_WIND = VGH
0172    C
0173          INIT_POSITION(1) = ((RADIUS_ERTH)*COS(XLATSNG))*SIN(XLNGSNG)
0174          INIT_POSITION(2) = ((RADIUS_ERTH)*COS(XLATSNG))*COS(XLNGSNG)
0175          INIT_POSITION(3) =  (RADIUS_ERTH)*SIN(XLATSNG)
0176    C
0177          CURR_POSITION(1) = INIT_POSITION(1)
0178          CURR_POSITION(2) = INIT_POSITION(2)
0179          CURR_POSITION(3) = INIT_POSITION(3)
0180    C
0181          CALL GRNRGE(INIT_POSITION,CURR_POSITION,AE,BE,RADIUS_ERTH,
0182       &                  2,2,GRD_RANGE)
0183    C
0184    C     PLACE VALUES IN COMMON BLOCK FOR WRITING TO EXTERNAL PLOT FILE
0185    C
0186          TIME_ARRAY(1)      = INIT_INTEGRAT_LMT
0187          LAT_ARRAY(1)       = INIT_LAT
0188          LON_ARRAY(1)       = INIT_LON
0189          ALT_ARRAY(1)       = INITIAL_ALT
0190          GRANGE_ARRAY(1)    = GRD_RANGE
0191          WINDAZ_ARRAY(1)    = WIND_AZIMUTH
0192          WIND_VEL_ARRAY(1) = WIND_VEL
0193    C
0194    C     PLACE VALUES IN COMMON BLOCK FOR MAKING MAP
0195    C
0196          TOFTAB(1) = INIT_INTEGRAT_LMT
0197          LNGTAB(1) = -1.0 * INIT_LON
0198          LATTAB(1) = INIT_LAT
0199          ALTTAB(1) = INITIAL_ALT
0200          JMPTAB(1) = INT2(0)
0201    C
0202    C     THE FOLLOWING WRITE STATEMENT PROVIDES HEADERS FOR THE
0203    C     OUTPUT FILES.
0204    C
```

217

```
0205              WRITE(14,30)'TIME',TAB,'TIME STEP',TAB,'ALTITUDE',TAB,
0206            &   'GRANGE',TAB,'WIND_AZIMUTH',TAB,'WIND_VEL',TAB,'LATITUDE',TAB,
0207            &   'LONGITUDE',TAB,'JUMP'
0208       30   FORMAT(2X,A,A1,A,A1,A,A1,A,A1,A,A1,A,A1,A,A1,A,A1,A)
0209     C
0210     C    WRITE THE INITIAL OUTPUT AT START TIME OF THE SIMULATION.
0211     C
0212              WRITE(14,20)INIT_INTEGRAT_LMT,TAB,INTEGRAT_STEP,TAB,
0213            &    INITIAL_ALT,TAB,GRD_RANGE,TAB,WIND_AZIMUTH,TAB,
0214            &    WIND_VEL,TAB,INIT_LAT,TAB,INIT_LON,TAB,JMPTAB(1)
0215     C
0216              INTEGRAT_STEP = FLIGHT_TIME / 2047.0
0217     C
0218              IF (INTEGRAT_STEP .GT. 10000.0) THEN
0219                INTEGRAT_STEP = 10000.0
0220              ELSEIF (INTEGRAT_STEP .LT. 1.0) THEN
0221                INTEGRAT_STEP = 1.0
0222              ENDIF
0223     C
0224              PERM_INT_STEP = INTEGRAT_STEP
0225              NEW_LAT = SNGL(INIT_LAT)
0226              NEW_LON = SNGL(INIT_LON)
0227     C
0228     C    READ IN ONE SET OF DATA POINTS OF THE ASCENT VELOCITY PROFILE.
0229     C
0230              READ (21,*) INIT_SEG_TIME, INIT_SEG_VEL
0231       100 READ (21,*,END = 101) FIN_SEG_TIME, FIN_SEG_VEL
0232              SegmentsFinished = .FALSE.
0233              GO TO 102
0234       101 FIN_SEG_TIME = FLIGHT_TIME
0235              FIN_SEG_VEL  = 0.0
0236              INIT_SEG_VEL = 0.0
0237              SegmentsFinished = .TRUE.
0238       102 CONTINUE
0239     C
0240     C    COMPUTE THE SLOPE AND Y INTERCEPT USED IN THE FORMULA FOR THE
0241     C    EQUATION OF A LINE.  THESE COMPONENTS COMPOSE THE FUNCTION USED
0242     C    IN DEVELOPING THE PROFILE.
0243     C
0244              SLOPE = ( FIN_SEG_VEL - INIT_SEG_VEL ) /
0245            &          ( FIN_SEG_TIME - INIT_SEG_TIME )
0246              Y_INTERCEPT = ( FIN_SEG_VEL - ( SLOPE * FIN_SEG_TIME ) )
0247              INIT_INTEGRAT_LMT = INIT_SEG_TIME
0248              FIN_INTEGRAT_LMT = INIT_SEG_TIME + INTEGRAT_STEP
0249     C
0250              IF(DIFFGT .GT. TOLERANCE)THEN
0251                FIN_INTEGRAT_LMT  = DIFFGT + INIT_SEG_TIME
0252                INTEGRAT_STEP = DIFFGT
0253                DIFFGT = 0.0
0254              ENDIF
0255     C
0256     C    IF THE UPPER LIMIT, BASED ON THE TIMESTEP, IS GREATER THAN THE
0257     C    FUNCTION BOUNDARY OF THE FUNCTION, RESET THE UPPER LIMIT TO
0258     C    COMPENSATE FOR THE UPPER BOUNDARY.
0259     C
0260              IF(FIN_INTEGRAT_LMT .GT. FIN_SEG_TIME)THEN
0261                DIFFGT = FIN_INTEGRAT_LMT - FIN_SEG_TIME
0262                FIN_INTEGRAT_LMT = FIN_INTEGRAT_LMT - DIFFGT
0263                INTEGRAT_STEP = FIN_INTEGRAT_LMT - INIT_INTEGRAT_LMT
0264              ENDIF
0265     C
0266              IF(FIN_INTEGRAT_LMT .GE. FLIGHT_TIME)THEN
0267                FIN_INTEGRAT_LMT  = FLIGHT_TIME
0268                INTEGRAT_STEP = FIN_INTEGRAT_LMT - INIT_INTEGRAT_LMT
```

```
0269              DIFFGT = 0.0
0270           ENDIF
0271   C
0272   C      USE FIRST TRAJECTORY INPUT AS 'RESET' VALUE.  THEN CHANGE MODE TO
0273   C      'OPERATE' AND CYCLE THROUGH REMAINDER OF TRAJECTORY.
0274   C
0275   C      THE FUNINT (FUNCTION INTEGRATION) ROUTINE WILL PERFORM THE
0276   C      INTEGRATION NEEDED TO PROVIDE THE DISTANCE TRAVELED (SEGMENT_DIST)
0277   C      IN TIME DEFINED BY THE FUNCTION.
0278   C
0279     200 IF(.NOT.SegmentsFinished)THEN
0280              CALL FUNINT(INIT_INTEGRAT_LMT,FIN_INTEGRAT_LMT,2,FUNCNT,ICHECK,
0281          &               SEGMENT_DIST,SLOPE,Y_INTERCEPT)
0282           ENDIF
0283   C
0284   C      COMPUTE THE CUMULATIVE DISTANCE TRAVELED BY THE BALLOON.
0285   C
0286           TOTAL_SEG_DIST = TOTAL_SEG_DIST + SEGMENT_DIST
0287           FINAL_ALT = TOTAL_SEG_DIST + INITIAL_ALT
0288   C
0289   C      NOGAPS/GRAMRT DERIVES THE WIND COMPONENTS, EAST (UGH (M/S))
0290   C      AND NORTH (VGH (M/S)).  THESE COMPONENTS ARE BASED UPON THE
0291   C      LATITUDE (NEW_LAT (DEG)/LAT_RAD (RAD)), LONGITUDE (NEW_LON
0292   C      (DEG)/LON_RAD (RAD)), AND ALTITUDE (FINAL_ALT (M)/HSIM (FEET)).
0293   C
0294           IF(CLIMATE .EQ. 13)THEN
0295             CALL NOGAPS(FINAL_ALT, NEW_LAT, NEW_LON, UGH, VGH, FIRST)
0296           ELSE
0297             HSIM    = FINAL_ALT * 3.28084
0298             CALL GRAMRT (FIRST)
0299           ENDIF
0300   C
0301   C      CONVERT THE WIND COMPONENTS TO A DISTANCE.
0302   C
0303           DIST_EAST = ((UGH + INIT_EAST_WIND)/2.0) * INTEGRAT_STEP
0304           DIST_NORTH = ((VGH + INIT_NORTH_WIND)/2.0) * INTEGRAT_STEP
0305   C
0306           DIST_EAST = DIST_EAST + TOTAL_EAST
0307           DIST_NORTH = DIST_NORTH + TOTAL_NORTH
0308   C
0309           TOTAL_EAST = DIST_EAST
0310           TOTAL_NORTH = DIST_NORTH
0311   C
0312           INIT_EAST_WIND = UGH
0313           INIT_NORTH_WIND = VGH
0314   C
0315           IF (VGH.EQ.0.0) THEN
0316             WIND_AZIMUTH = 90.0
0317           ELSE
0318             WIND_AZIMUTH = (ATAN(UGH/VGH))*R2D
0319           ENDIF
0320           IF( VGH.LT.0.0) THEN
0321             WIND_AZIMUTH = 180.0+WIND_AZIMUTH
0322           ELSEIF( UGH.LT.0.0 ) THEN
0323             WIND_AZIMUTH = 360.0+WIND_AZIMUTH
0324           ENDIF
0325   C
0326           WIND_VEL = SQRT((UGH**2) + (VGH**2)
0327   C
0328           DELTAT = INTEGRAT_STEP + DELTAT
0329   C
0330           WCMAG = SQRT((DIST_EAST**2) + (DIST_NORTH**2))
0331   C
0332   C      THE LATITUDE/LONGITUDE RESOLUTION IS 1.0 METER (ANGLE_RESOLUTION).
```

```
0333    C
0334            IF(WCMAG .GT. ANGLE_RESOLUTION)THEN
0335    C
0336            TOTAL_EAST  = 0.0
0337            TOTAL_NORTH = 0.0
0338    C
0339            LAT_DEG = LAT_RAD * R2D
0340            LON_DEG = LON_RAD * R2D
0341    C
0342    C       SUBROUTINE ECOORD USES THE LATITUDE (LAT_DEG), LONGITUDE (LON_DEG),
0343    C       AS WELL AS THE DISTANCE THE BALLOON TRAVELED DURING THE TIMESTEP
0344    C       (DIST_EAST, DIST_NORTH) TO PRODUCE THE NEW LATITUDE AND LONGITUDE
0345    C       (NEW_LAT, NEW_LON).  IT ALSO USES THE CONSTANT RADIUS_ERTH AND THE FLAG
0346    C       IS. THIS FLAG ALLOWS FOR A NORTH AND EAST RANGE INSTEAD OF
0347    C       MAGNITUDE AND DIRECTION.
0348    C
0349            CALL ECOORD (LAT_DEG,LON_DEG,DIST_EAST,DIST_NORTH,
0350           .             RADIUS_ERTH,IS,NEW_LAT,NEW_LON)
0351    C
0352    C       LON_ERTH_ROT = (ERTH_ROTATION * DELTAT) * R2D
0353    C       NEW_LON = NEW_LON + LON_ERTH_ROT
0354            IF (NEW_LON.GT.360.0) THEN
0355              NEW_LON = NEW_LON - 360.0
0356            ELSE IF (NEW_LON.LT.-360.0) THEN
0357              NEW_LON = NEW_LON + 360.0
0358            ENDIF
0359            DELTAT = 0.0
0360    C
0361            DIFLAT = ABS(NEW_LAT - LAT_DEG)
0362            DIFLON = ABS(NEW_LON - LON_DEG)
0363    C
0364            NEW_LAT = SNGL(NEW_LAT)
0365            NEW_LON = SNGL(NEW_LON)
0366            LAT_RAD = NEW_LAT/R2D
0367            LON_RAD = NEW_LON/R2D
0368            XLATSNG = SNGL(LAT_RAD)
0369            XLNGSNG = SNGL(LON_RAD)
0370    C
0371            CURR_POSITION(1) = ((RADIUS_ERTH)*COS(XLATSNG))*SIN(XLNGSNG)
0372            CURR_POSITION(2) = ((RADIUS_ERTH)*COS(XLATSNG))*COS(XLNGSNG)
0373            CURR_POSITION(3) = (RADIUS_ERTH)*SIN(XLATSNG)
0374    C
0375            CALL GRNRGE(INIT_POSITION,CURR_POSITION,AE,BE,RADIUS_ERTH,
0376           &            2,2,GRD_RANGE)
0377    C
0378            ENDIF
0379    C
0380            T2TOF = ABS(FIN_INTEGRAT_LMT - FLIGHT_TIME)
0381            PERCNT_DONE = JNINT((FIN_INTEGRAT_LMT/FLIGHT_TIME)*100.0)
0382
0383            IF( PERCNT_DONE .GT. last_value) THEN
0384              WRITE(CHAR_DATA,4) PERCNT_DONE
0385      4       format ( i3 , ' %' )
0386              PRTEXT = CHAR_DATA
0387              CALL SetiText( %val(ItHndl), %val(PrText) )
0388              last_value = PERCNT_DONE
0389            ENDIF
0390    C
0391            IF(T2TOF .LE. 0.5)GOTO 789
0392    C
0393    C       IF THE DIFFERENCE BETWEEN THE TIMESTEP AND SEGMENT IS GREATER THAN
0394    C       ZERO, READ IN ANOTHER SEGMENT OF THE ASCENT PROFILE.
0395    C
0396            IF( DIFFGT .GT. TOLERANCE )THEN
```

```
0397              INIT_SEG_TIME = FIN_SEG_TIME
0398              INIT_SEG_VEL  = FIN_SEG_VEL
0399            GOTO 100
0400          ENDIF
0401     C
0402          IF( INTEGRAT_STEP .EQ. DIFFNC .AND. DIFFTM .GT. TOLERANCE) THEN
0403            INTEGRAT_STEP = DIFFTM
0404            INIT_SEG_TIME = FIN_SEG_TIME
0405            INIT_SEG_VEL  = FIN_SEG_VEL
0406            DIFFNC = 0.0
0407            DIFFTM = 0.0
0408            GOTO 100
0409          ENDIF
0410     C
0411          INTEGRAT_STEP = PERM_INT_STEP
0412     C
0413          NO_OF_PTS = NO_OF_PTS + 1
0414          IF(NO_OF_PTS .GT. 2048.0)GOTO 100
0415     C
0416          TIME_ARRAY(NO_OF_PTS)     = FIN_INTEGRAT_LMT
0417          LAT_ARRAY(NO_OF_PTS)      = NEW_LAT
0418          LON_ARRAY(NO_OF_PTS)      = NEW_LON
0419          ALT_ARRAY(NO_OF_PTS)      = FINAL_ALT
0420          GRANGE_ARRAY(NO_OF_PTS)   = GRD_RANGE
0421          WINDAZ_ARRAY(NO_OF_PTS)   = WIND_AZIMUTH
0422          WIND_VEL_ARRAY(NO_OF_PTS) = WIND_VEL
0423     C
0424          TOFTAB(NO_OF_PTS) = FIN_INTEGRAT_LMT
0425          LATTAB(NO_OF_PTS) = NEW_LAT
0426          ALTTAB(NO_OF_PTS) = FINAL_ALT
0427     C
0428     C    CONVERT PROGRAM'S INTERNAL WEST LONGITUDE TO EAST LONGTUDE FOR MAP
0429          IF(NEW_LON.GT.180.0) THEN
0430            LNGTAB(NO_OF_PTS) = 360.0 - NEW_LON
0431            IF(OFF_WEST)THEN
0432              JMPTAB(NO_OF_PTS) = INT2(0)
0433            ELSE
0434              JMPTAB(NO_OF_PTS) = INT2(-1)
0435              OFF_WEST = .TRUE.
0436            ENDIF
0437          ELSE IF(NEW_LON.LT.-180.0) THEN
0438            LNGTAB(NO_OF_PTS) = 360.0 + NEW_LON
0439            IF(OFF_EAST)THEN
0440              JMPTAB(NO_OF_PTS) = INT2(0)
0441            ELSE
0442              JMPTAB(NO_OF_PTS) = INT2(1)
0443              OFF_EAST = .TRUE.
0444            ENDIF
0445          ELSE
0446            LNGTAB(NO_OF_PTS) = -1.0 * NEW_LON
0447            IF (OFF_WEST) THEN
0448              JMPTAB(NO_OF_PTS) = INT2(1)
0449              OFF_WEST = .FALSE.
0450            ELSE IF (OFF_EAST) THEN
0451              JMPTAB(NO_OF_PTS) = INT2(-1)
0452              OFF_EAST = .FALSE.
0453            ELSE
0454              JMPTAB(NO_OF_PTS) = INT2(0)
0455            ENDIF
0456          ENDIF
0457     C
0458          WRITE(14,20)FIN_INTEGRAT_LMT,TAB,INTEGRAT_STEP,TAB,
0459         &           FINAL_ALT,TAB,GRD_RANGE,TAB,WIND_AZIMUTH,TAB,WIND_VEL,
0460         &           TAB,NEW_LAT,TAB,NEW_LON,TAB,JMPTAB(NO_OF_PTS)
```

221

```
0461          20 FORMAT (F9.2,A1,F8.2,A1,F10.3,A1,F15.3,A1,2(F8.3,A1),
0462           &            F7.4,A1,F8.4,A1,I2)
0463      C
0464      C     DETERMINE THE POSITION OF THE CUMULATIVE TIMESTEP ITERATIONS
0465      C     IN RELATION TO THE END POINT OF THE ASCENT PROFILE SEGMENT.
0466      C
0467            DIFFNC = ABS(FIN_INTEGRAT_LMT - FIN_SEG_TIME)
0468            DIFFTM = ABS(DIFFNC - INTEGRAT_STEP)
0469      C
0470            IF( DIFFNC .LT. TOLERANCE )THEN
0471              INIT_SEG_TIME = FIN_SEG_TIME
0472              INIT_SEG_VEL  = FIN_SEG_VEL
0473              GOTO 100
0474            ELSEIF( DIFFNC .GE. INTEGRAT_STEP .OR. DIFFTM .LT.TOLERANCE )THEN
0475              INIT_INTEGRAT_LMT = FIN_INTEGRAT_LMT
0476              FIN_INTEGRAT_LMT  = FIN_INTEGRAT_LMT + INTEGRAT_STEP
0477              GO TO 200
0478            ELSE
0479              INTEGRAT_STEP      = DIFFNC
0480              INIT_INTEGRAT_LMT = FIN_INTEGRAT_LMT
0481              FIN_INTEGRAT_LMT  = FIN_SEG_TIME
0482              GOTO 200
0483            ENDIF
0484      C
0485        789 NO_OF_PTS = NO_OF_PTS + 1
0486      C
0487            TIME_ARRAY(NO_OF_PTS)       = FIN_INTEGRAT_LMT
0488            LAT_ARRAY(NO_OF_PTS)        = NEW_LAT
0489            LON_ARRAY(NO_OF_PTS)        = NEW_LON
0490            ALT_ARRAY(NO_OF_PTS)        = FINAL_ALT
0491            GRANGE_ARRAY(NO_OF_PTS)     = GRD_RANGE
0492            WINDAZ_ARRAY(NO_OF_PTS)     = WIND_AZIMUTH
0493            WIND_VEL_ARRAY(NO_OF_PTS) = WIND_VEL
0494      C
0495            ntrpts = NO_OF_PTS
0496            TOFTAB(NO_OF_PTS) = FIN_INTEGRAT_LMT
0497            LATTAB(NO_OF_PTS) = NEW_LAT
0498            ALTTAB(NO_OF_PTS) = FINAL_ALT
0499      C
0500      C     CONVERT PROGRAM'S INTERNAL WEST LONGITUDE TO EAST LONGTUDE FOR MAP
0501            IF(NEW_LON.GT.180.0) THEN
0502              LNGTAB(NO_OF_PTS) = 360.0 - NEW_LON
0503              IF(OFF_WEST)THEN
0504                JMPTAB(NO_OF_PTS) = INT2(0)
0505              ELSE
0506                JMPTAB(NO_OF_PTS) = INT2(-1)
0507                OFF_WEST = .TRUE.
0508              ENDIF
0509            ELSE IF(NEW_LON.LT.-180.0) THEN
0510              LNGTAB(NO_OF_PTS) = 360.0 + NEW_LON
0511              IF(OFF_EAST)THEN
0512                JMPTAB(NO_OF_PTS) = INT2(0)
0513              ELSE
0514                JMPTAB(NO_OF_PTS) = INT2(1)
0515                OFF_EAST = .TRUE.
0516              ENDIF
0517            ELSE
0518              LNGTAB(NO_OF_PTS) = -1.0 * NEW_LON
0519              IF (OFF_WEST) THEN
0520                JMPTAB(NO_OF_PTS) = INT2(1)
0521                OFF_WEST = .FALSE.
0522              ELSE IF (OFF_EAST) THEN
0523                JMPTAB(NO_OF_PTS) = INT2(-1)
0524                OFF_EAST = .FALSE.
```

```
0525            ELSE
0526               JMPTAB(NO_OF_PTS) = INT2(0)
0527             ENDIF
0528          ENDIF
0529    C
0530          call SetPort( %val(SavedPort) )
0531          call DisposDialog( %val(StatPtr) )
0532    C
0533    C     CLOSE TEMP FILES
0534    C
0535          CLOSE(UNIT=IOTEM1,STATUS='DELETE')
0536          CLOSE(UNIT=IOTEM2)
0537    C
0538    C     CLOSE OUTPUT FILE
0539    C
0540          CLOSE(UNIT=14)
0541    C
0542          RETURN
0543          END
0544    C
0545          FUNCTION FUNCNT(T,SLOPE,YINTER)
0546          REAL*4 SLOPE,YINTER,FUNCNT
0547          REAL*8 T
0548    C
0549          FUNCNT = ( SLOPE * T ) + YINTER
0550    C
0551          RETURN
0552          END
0553    C
0554          SUBROUTINE FUNINT (XL,XH,INTORD,FNAM,ICHECK,VOINT,P,Y)
0555    C
0556    C*******************************************************************C
0557    C                                                                  C
0558    C    PURPOSE  1) A FUNCTIONAL INTEGRATION ROUTINE USING A VARIABLE C
0559    C                   ORDER GAUSSIAN-LEGENDRE ALGORITHM.             C
0560    C                                                                  C
0561    C    INPUTS    DESCRIPTION                                        C
0562    C                                                                  C
0563    C       ( XL )   THE LOWER BOUND ON THE INTEGRAL (CAN BE REAL*8)  C
0564    C       ( XH )   THE UPPER BOUND ON THE INTEGRAL (CAN BE REAL*8)  C
0565    C    (INTORD) ORDER OF THE GAUSS-LEGENDRE POLYNOMIAL: 2-10        C
0566    C               INCLUSIVE AND 12, 16, 20, 24 AND 32 ARE IMPLEMENTED.  C
0567    C       (FNAM)   NAME OF THE FORTRAN EXTERNAL FUNCTION            C
0568    C                TO BE INTEGRATED (CAN BE REAL*8)                 C
0569    C                                                                  C
0570    C       OUTPUTS  DESCRIPTION                                      C
0571    C                                                                  C
0572    C    (VOINT)  THE VALUE OF THE INTEGRAL BETWEEN THE SPECIFIED LIMITS  C
0573    C    (ICHECK) CHECK FOR: PROPER EXECUTION = 0                     C
0574    C                FAULT FOUND       = -1                           C
0575    C                                                                  C
0576    C    PROGRAM REQUIREMENTS                                         C
0577    C                                                                  C
0578    C    THE PROGRAM ASSUMES THAT A FORTRAN EXTERNAL FUNCTION (WHICH  C
0579    C        MAY BE REAL*8) IS DEFINED SOMEWHERE WITHIN THE CALLING CODE.  C
0580    C    ( SINCE IT IS AN EXTERNAL FUNCTION IT NEED NOT NECESSARILY BE  C
0581    C    DEFINED IN THE ROUTINE THAT MAKES THE SUBROUTINE CALL ).     C
0582    C                                                                  C
0583    C       CAVEATS                                                   C
0584    C                                                                  C
0585    C       THIS IS A VERY SIMPLE ROUTINE WHICH NEVERTHELESS IS VERY  C
0586    C       FAST (THERE ARE ONLY INTORD EVALUATIONS OF THE FUNCTION)  C
0587    C       AND QUITE ACCURATE FOR WELL BEHAVED FUNCTIONS.            C
0588    C       AS WITH ANY NUMERICAL TECHNIQUE FUNCTIONS WITH SINGULARITIES  C
```

```
0589    C        AND PERIODIC FUNCTIONS INTEGRATED OVER MANY PERIODS ARE OFTEN   C
0590    C        NOT HANDLED WELL.                                               C
0591    C        IN PRINCIPLE THE ACCURACY OF THE ALGORITHM INCREASES WITH THE   C
0592    C        THE ORDER OF THE POLYNOMIAL USED, HOWEVER ROUNDOFF ERRORS WILL C
0593    C        LIMIT THIS TREND.                                               C
0594    C                                                                        C
0595    C=======================================================================C
0596    C
0597             PARAMETER (MIN = 2, MAX = 32)
0598             REAL*8 WEIGHT(MIN:MAX,MAX/2), Z(MIN:MAX,MAX/2)
0599             REAL*8 SUMH, DELTAH, HALF, TERM
0600             REAL*4 VOINT, XH, XL, FNAM
0601             CHARACTER*1 SR
0602             DATA HALF / 0.5 /
0603    C
0604    C        DATA STATEMENTS CONTAINING THE WEIGHTS AND ZEROES OF THE
0605    C        LEGENDRE POLYNOMIALS
0606    C
0607             DATA WEIGHT(2,1) / 1. /
0608             DATA (WEIGHT(3,I),I=1,2) / .555555555555556,.888888888888889 /
0609             DATA (WEIGHT(4,I),I=1,2) / .347854845137454,.652145154862546 /
0610             DATA (WEIGHT(5,I),I=1,3) / .236926885056189,.478628670499366,
0611           &     .568888888888889 /
0612             DATA (WEIGHT(6,I),I=1,3) / .171324492379170,.360761573048139,
0613           &     .467913934572691 /
0614             DATA (WEIGHT(7,I),I=1,4) / .129484966168870,.279705391489277,
0615           &     .381830050505119,.417959183673469 /
0616             DATA (WEIGHT(8,I),I=1,4) / .101228536290376,.222381034453374,
0617           &     .313706645877887,.362683783378362 /
0618             DATA (WEIGHT(9,I),I=1,5) / .081274388361574,.180648160694857,
0619           &     .260610696402935,.312347077040003,.330239355001260 /
0620             DATA (WEIGHT(10,I),I=1,5) / .066671344308688,.149451349150581,
0621           &     .219086362515982,.269266719309996,.295524224714753 /
0622             DATA (WEIGHT(12,I),I=1,6) / .047175336386512,.106939325995318,
0623           &     .160078328543346,.203167426723066,.233492536538355,
0624           &     .249147045813403 /
0625             DATA (WEIGHT(16,I),I=1,8) / .027152459411754,.062253523938648,
0626           &     .095158511682493,.124628971255534,.149595988816577,
0627           &     .169156519395003,.182603415044924,.189450610455068 /
0628             DATA (WEIGHT(20,I),I=1,10) / .017614007139152,.040601429800387,
0629           &     .062672048334109,.083276741576705,.101930119817240,
0630           &     .118194531961518,.131688638449177,.142096109318382,
0631           &     .149172986472604,.152753387130726 /
0632             DATA (WEIGHT(24,I),I=1,12) / .012341229799987,.028531388628934,
0633           &     .044277438817420,.059298584915437,.073346481411080,
0634           &     .086190161531953,.097618652104114,.107444270115966,
0635           &     .115505668053726,.121670472927803,.125837456346828,
0636           &     .127938195346752 /
0637             DATA (WEIGHT(32,I),I=1,16) / .007018610009470,.016274394730906,
0638           &     .025392065309262,.034273862913021,.042835898022227,
0639           &     .050998059262376,.058684093478536,.065822222776362,
0640           &     .072345794108849,.078193895787070,.083311924226947,
0641           &     .087652093004404,.091173878695764,.093844399080805,
0642           &     .095638720079274,.096540088514728 /
0643    C
0644             DATA Z(2,1)              / .577350269189626
0645             DATA (Z(3,I),I=1,2) / .774596669241483,.0
0646             DATA (Z(4,I),I=1,2) / .861136311594053,.339981043584856
0647             DATA (Z(5,I),I=1,3) / .906179845938664,.538469310105683,.0
0648             DATA (Z(6,I),I=1,3) / .932469514203152,.661209386466265,
0649           &     .238619186083197 /
0650             DATA (Z(7,I),I=1,4) / .949107912342759,.741531185599394,
0651           &     .405845151377397,.0 /
0652             DATA (Z(8,I),I=1,4) / .960289856497536,.796666477413627,
```

```
0653        &      .525532409916329,.183434642495650 /
0654         DATA (Z(9,I),I=1,5) / .968160239507626,.836031107326636,
0655        &      .613371432700590,.324253423403809,.0 /
0656         DATA (Z(10,I),I=1,5) / .973906528517172,.865063366688985,
0657        &      .679409568299024,.433395394129247,.148874338981631 /
0658         DATA (Z(12,I),I=1,6) / .981560634246719,.904117256370475,
0659        &      .769902674194305,.587317954286617,.367831498998180,
0660        &      .125233408511469 /
0661         DATA (Z(16,I),I=1,8) / .989400934991650,.944575023073233,
0662        &      .865631202387832,.755404408355003,.617876244402644,
0663        &      .458016777657227,.281603550779259,.095012509837637 /
0664         DATA (Z(20,I),I=1,10) / .993128599185095,.963971927277914,
0665        &      .912234428251326,.839116971822219,.746331906460151,
0666        &      .636053680726515,.510867001950827,.373706088715420,
0667        &      .227785851141645,.076526521133497 /
0668         DATA (Z(24,I),I=1,12) / .995187219997021,.974728555971309,
0669        &      .938274552002733,.886415527004401,.820001985973903,
0670        &      .740124191578554,.648093651936976,.545421471388840,
0671        &      .433793507626045,.315042679696163,.191118867473616,
0672        &      .064056892862686 /
0673         DATA (Z(32,I),I=1,16) / .997263861849481,.985611511545268,
0674        &      .964762255587506,.934906075937740,.896321155766052,
0675        &      .849367613732570,.794483795967942,.732182118740290,
0676        &      .663044266930215,.587715757240762,.506899908932229,
0677        &      .421351276130635,.331868602282128,.239287362252137,
0678        &      .144471961582796,.048307665687738 /
0679   C
0680        VOINT  = 0.
0681        ICHECK = 0
0682   C
0683   C     CHECK THAT THE ORDER OF INTEGRATION IS IN BOUNDS
0684   C
0685        IF ( (INTORD .LT. MIN) .OR. (INTORD .GT. MAX) ) THEN
0686          PRINT*, ' INTEGRATION OF THAT ORDER IS NOT AVAILABLE. '
0687          ICHECK = -1
0688          RETURN
0689        END IF
0690   C
0691   C     CHECK THAT THE ORDER OF INTEGRATION IS AVAILABLE
0692   C
0693        IF ( (WEIGHT(INTORD, 1) .EQ. 0.) ) THEN
0694          PRINT*, ' INTEGRATION OF THAT ORDER IS NOT AVAILABLE. '
0695          ICHECK = -1
0696          RETURN
0697        END IF
0698   C
0699   C     CALCULATE SEVERAL NECESSARY QUANITIIES
0700   C
0701        DELTAH = (XH - XL) * HALF
0702        SUMH   = (XH + XL) * HALF
0703        NSTEP = INTORD/2
0704   C
0705   C     INTEGRATE THE FUNCTION WITH A INTORD ORDER ALGORITHM
0706   C
0707        DO I = 1,NSTEP
0708          TERM = FNAM ( SUMH + DELTAH * Z (INTORD,I),F,Y)
0709          VOINT = VOINT + WEIGHT (INTORD,I) * TERM
0710          TERM = FNAM ( SUMH - DELTAH * Z (INTORD,I),F,Y)
0711          VOINT = VOINT + WEIGHT (INTORD,I) * TERM
0712        END DO
0713   C
0714   C     MULTIPLY BY LIMITS CONSTANT AND TAKE CARE OF ODD POINT FOR INTORD ODD
0715   C
0716        IF ( NSTEP*2 .EQ. INTORD ) THEN
```

```
0717              VOINT = DELTAH * VOINT
0718            ELSE
0719              VOINT = ( VOINT + WEIGHT (INTORD, NSTEP + 1)*FNAM(SUMH,P,Y))*
0720         &            DELTAH
0721            END IF
0722   C
0723            RETURN
0724            END
0725   C
0726            SUBROUTINE ECOORD ( ALATR, ALONR, DRP, URP, RE, IS, ALATP, ALONP )
0727   C
0728   C=========================================================================C
0729   C                                                                         C
0730   C     PURPOSE    1) DETERMINE LATITUDE (ALATP) AND LONGITUDE (ALONP)      C
0731   C                   OF A DESIRED POINT USING REFERENCE LATITUDE           C
0732   C                   (ALATR) AND LONGITUDE (ALONR).                        C
0733   C                2) INPUTS  ARE INTERPRETED DIFFERENTLY BY VALUE OF       C
0734   C                   FLAG (IS)                                             C
0735   C                      (IS) = 0  (DRP) RANGE FROM REFERENCE POINT         C
0736   C                                (+) EASTERLY TO MERIDIAN OF DESIRED      C
0737   C                                POINT                                    C
0738   C                                (URP) RANGE ALONG MERIDIAN (+) UP TO     C
0739   C                                DESIRED POINT                            C
0740   C                      (IS) = 1  (DRP) RANGE FROM REFERENCE POINT TO      C
0741   C                                DESIRED POINT                            C
0742   C                                (URP) AZIMUTH OF RANGE LINE-OF-SITE TO   C
0743   C                                DESIRED POINT                            C
0744   C                                                                         C
0745   C                    NOTE - RANGES ARE ALONG EARTH SURFACE AND            C
0746   C                           AZIMUTH IS IN DEGREES FROM NORTH              C
0747   C                           LATITUDES ARE GEOCENTRIC                      C
0748   C                                                                         C
0749   C     INPUTS     DESCRIPTION                                              C
0750   C                                                                         C
0751   C     (ALATR)    LATITUDE OF REFERENCE POINT (DEG)                        C
0752   C     (ALONR)    LONGITUDE (+) EAST OF GREENWICH OF REFERENCE POINT       C
0753   C                (DEG)                                                    C
0754   C     (DRP)      RANGE FROM REFERENCE POINT TO                            C
0755   C                   (IS) = 0 (+) EAST ALONG LATITUDE TO MERIDIAN OF       C
0756   C                           DESIRED POINT                                 C
0757   C                   (IS) = 1 DESIRED POINT                                C
0758   C     (URP)      (IS) = 0 RANGE (+) NORTH ALONG MERIDIAN OF DESIRED       C
0759   C                         POINT FROM REFERENCE LATITUDE TO DESIRED        C
0760   C                         POINT                                          C
0761   C                (IS) = 1 AZIMUTH OF RANGE LINE-OF-SITE FROM REFERENCE    C
0762   C                         POINT TO DESIRED POINT                          C
0763   C     (RE)       EARTH RADIUS AT REFERENCE POINT                          C
0764   C     (IS)       FLAG TO INTERPRET THE MEANINGS OF (DRP) AND (URP)        C
0765   C                                                                         C
0766   C     OUTPUTS    DESCRIPTION                                              C
0767   C                                                                         C
0768   C     (ALATP)    LATITUDE OF DESIRED POINT (DEG)                          C
0769   C     (ALONP)    LONGITUDE OF DESIRED POINT (DEG)                         C
0770   C                                                                         C
0771   C  -  -   -     -   -  -   -   -   -   -   -  -   -   -   -   -   -  C
0772   C                                                                         C
0773   C     REQUIREMENTS 1) SINE ROUTINE         - SIN                          C
0774   C                  2) COSINE ROUTINE       - COS                          C
0775   C                  3) SQUARE ROOT ROUTINE - SQRT                          C
0776   C                  4) ARCSINE ROUTINE      - ASIN                         C
0777   C                                                                         C
0778   C=========================================================================C
0779   C
0780          DATA R2D / 57.29577951 /
```

```
0781            REAL*4 ALATR,ALONR,DRP,URP,ALATP,ALONP
0782      C
0783            ALONR = ALONR * -1.0
0784      C
0785            IF( RE ) 50, 50, 10
0786       10 CONTINUE
0787            THD = DRP/RE
0788      C
0789            IF( IS ) 20, 20, 30
0790      C
0791       20 CONTINUE
0792            THU = URP/RE
0793            TT  = COS ( THU )
0794            T1  = SIN ( THU )
0795            T2  = COS ( THD )*TT
0796            T3  = SIN ( THD )*TT
0797            GO TO 40
0798      C
0799       30 CONTINUE
0800            THU = URP/R2D
0801            TT  = SIN ( THD )
0802            T1  = COS ( THU )*TT
0803            T2  = COS ( THD )
0804            T3  = SIN ( THU )*TT
0805      C
0806       40 CONTINUE
0807            TT  = ALATR/R2D
0808            SPP = COS ( TT )*T1 + SIN ( TT )*T2
0809            IF( SPP .GT.  1.0 ) SPP = 1.
0810            IF( SPP .LT. -1.0 ) SPP =-1.
0811            SLP = SQRT ( 1. - SPP*SPP )
0812            IF( SLP .NE.  0.0 ) SLP = T3/SLP
0813            IF( SLP .LT. -1.0 ) SLP =-1.
0814            IF( SLP .GT.  1.0 ) SLP = 1.
0815      C
0816            ALATP = ASIN ( SPP )*R2D
0817            ALONP = ASIN ( SLP )*R2D + ALONR
0818       50 CONTINUE
0819      C
0820            ALONR = ALONR * -1.0
0821            ALONP = ALONP * -1.0
0822            RETURN
0823            END
0824      C
0825            SUBROUTINE OPENIT(XLATIN,XLONIN,INFILE,TOTTOF,ICLIMT)
0826      C
0827      C.... THIS ROUTINE OPENS THE FILES NORMALLY ATTACHED WITH JCL DIRECTIVES
0828      C     ON THE GOULD VERSION
0829      C
0830      C.... WRITTEN  10/18/89   L SCHILLING   NASA/ADFRF.
0831      C
0832            CHARACTER*255 INFILE
0833            CHARACTER CLAT*3,CLON*4,CLIMT*2
0834            CHARACTER*7 CTOF
0835            INTEGER LAT,LON,TOF,ICLIMT
0836      C
0837            LAT = XLATIN
0838            LON = XLONIN
0839            TOF = TOTTOF
0840            WRITE(CLAT,'(I3)')LAT
0841            WRITE(CLON,'(I4)')LON
0842            WRITE(CTOF,'(I7)')TOF
0843            WRITE(CLIMT,'(I2)')ICLIMT
0844      C
```

227

```
0845            OPEN (UNIT=21,FILE=INFILE,STATUS='OLD',READONLY)
0846      C
0847            OPEN(14,FILE=CLAT//'_'//CLON//'_'//CTOF//'.'//CLIMT,
0848         .      FORM='FORMATTED',ACCESS='SEQUENTIAL', STATUS='NEW')
0849      C
0850            RETURN
0851            END
0852
0853      C.... THIS FILE CONTAINS SUBROUTINES USED BY REAL-TIME GRAM PROGRAM.
0854      C    1/24/89   L SCHILLING.  THE MAJORITY HAVE BEEN UNAFFECTED, BUT
0855      C    SEVERAL HAVE BEEN SIGNIFICANTLY MODIFIED FROM THE ORIGINAL
0856      C    GRAM88 CODE.
0857      C
0858            SUBROUTINE CORLAT(A,B,C,D,E,F,G,H,AI,AJ,AK,SP1,SP2,SD1,SD2,ST1,
0859           1 ST2,SU1,SU2,SV1,SV2,UD1,UD2,VD1,VD2,RD,RT,RV)
0860            IF(SD1*ST1*SD2*ST2*RD*RT*RV.GT.0.) GO TO 5
0861      C.....DEFAULT VALUES AVOID DIVISION BY ZERO
0862            IF(SD1.LE.0.) SD1=0.001
0863            IF(ST1.LE.0.) ST1=0.001
0864            IF(SD2.LE.0.) SD2=0.001
0865            IF(ST2.LE.0.) ST2=0.001
0866            IF(RD.LE.0.) RD = .00001
0867            IF(RT.LE.0.) RT = .00001
0868            IF(RV.LE.0.) RV = .00001
0869      5     CONTINUE
0870            IF (ABS(UD1).LE.0.) UD1 = 0.001
0871            IF (ABS(VD1).LE.0.) VD1 = 0.001
0872            IF (ABS(SU1).LE.0.) SU1 = 0.001
0873            IF (ABS(SV1).LE.0.) SV1 = 0.001
0874            IF (ABS(UD1).GE.1.) UD1 = 0.99*UD1/ABS(UD1)
0875            IF (ABS(VD1).GE.1.) VD1 = 0.99*VD1/ABS(VD1)
0876            A=RD*SD2/SD1
0877            B=SD2*SQRT(1-RD*RD)
0878            TD2=(SP2*SP2-SD2*SD2-ST2*ST2)/(2*SD2*ST2)
0879            TD1=(SP1*SP1-SD1*SD1-ST1*ST1)/(2*SD1*ST1)
0880            IF(ABS(TD1).LE.0.) TD1 = 0.001
0881            IF(ABS(TD2).LE.0.) TD2 = 0.001
0882            IF(ABS(TD2).GE.1.0) TD2=0.99*TD2/ABS(TD2)
0883            IF(ABS(TD1).GE.1.0) TD1=0.99*TD1/ABS(TD1)
0884            C=(ST2/ST1)*(RT-RD*TD2*TD1)/(1-TD1*TD1*RD*RD)
0885            D=(RT*ST2-C*ST1)/(A*TD1*SD1)
0886            E=     ST2*ST2-C*C*ST1*ST1-D*D*SD2*SD2-2*C*D*RD*TD1*ST1*SD2
0887            IF(E.GE.0.) GO TO 10
0888            E=0.
0889      10 E=SQRT(E)
0890            F=(SU2/SU1)*(RV-RD*UD2*UD1)/(1-RD*RD*UD1*UD1)
0891            G=(RV*SU2-F*SU1)/(RD*UD1*SD2)
0892            H=     SU2*SU2-F*F*SU1*SU1-G*G*SD2*SD2-2*F*G*RD*UD1*SD2*SU1
0893            IF(H.GE.0.) GO TO 15
0894            H=0.
0895      15 H=SQRT(H)
0896            AI=(SV2/SV1)*(RV-RD*VD2*VD1)/(1-RD*RD*VD1*VD1)
0897            AJ=(RV*SV2-AI*SV1)/(RD*VD1*SD2)
0898            AK=     SV2*SV2-AI*AI*SV1*SV1-AJ*AJ*SD2*SD2-2*AI*AJ*RD*VD1*SD2*SV1
0899            IF(AK.GE.0.) GO TO 25
0900            AK=0.
0901      25 AK=SQRT(AK)
0902            RETURN
0903            END
0904      C
0905            FUNCTION CORREL(X)
0906            DATA A,B/19.51615854016301,1.00041693941245578/
0907            RHO = 1./EXP(B*X)
0908            IF(X.LT.0.05)RHO = 1. - A*X**2
```

228

```
0909          CORREL = RHO
0910          RETURN
0911          END
0912    C
0913          SUBROUTINE FAIR (PG, DG, TG, PJ, DJ, TJ, IH, R, D, T, DPXG,
0914        $ DPYG, DPXJ, DPYJ, DPX, DPY, DTXG, DTYG, DTXJ, DTYJ, DTX, DTY)
0915    C.....FAIRS BETWEEN GROVES AND JACCHIA VALUES 90 LE HEIGHT LE 115 KM
0916          DIMENSION CZ(6)
0917    C.....FAIRING VALUES
0918          DATA CZ /1.0,0.9045085,0.6545085,0.3454915,0.0954915,0.0/
0919    C     HEIGHT INDEX
0920          I =(IH - 85)/5
0921    C     GROVES FAIRING COEFFICIENT
0922          CZI = CZ(I)
0923    C     JACCHIA FAIRING COEFFICIENT
0924          SZI = 1.0 - CZI
0925    C     FAIRED TEMPERATURE
0926          T = TG*CZI + TJ*SZI
0927    C     FAIRED DENSITY
0928          D = EXP(ALOG(DG)*CZI + ALOG(DJ)*SZI)
0929    C     FAIRED GAS CONSTANT
0930          RG = PG/(DG*TG)
0931          RJ = PJ/(DJ*TJ)
0932          R = RG*CZI + RJ*SZI
0933          P = R*D*T
0934          DPX = DPXG*CZI + DPXJ*SZI
0935    C     DP/DY FOR GEOSTROPHIC WINDS
0936          DPY=DPYG*CZI+DPYJ*SZI
0937          DTX = DTXG*CZI + DTXJ*SZI
0938    C      DT/DY FOR THERMAL WINDS
0939          DTY = DTYG * CZI + DTYJ * SZI
0940          RETURN
0941          END
0942    C
0943          SUBROUTINE GRAMIN (XLATIN,XLONIN,ALTINT,ICLIMT)
0944    C
0945    C.... THIS ROUTINE READS IN ALL DATA FILES TO INITIALIZE ARRAYS
0946    C     FOR USE IN GRAM PROGRAM.
0947    C
0948    C.... WRITTEN   24 JAN 89   L SCHILLING   NASA/ADFRF.
0949    C
0950          COMMON /C4    /   DUMMY4(2529), THET1, DUMMY5(2)
0951          COMMON /IOTEMP/ IOTEM1,IOTEM2,IUG    ,IUN    ,DD    ,XMJD   ,PHI1  ,
0952          .               PHI   ,NSAME ,RP1    ,RD1    ,RT1   ,SP1    ,SD1   ,
0953          .               ST1   ,RU1   ,RV1    ,SU1    ,SV1   ,MN     ,IDA   ,
0954          .               IYR   ,H1    ,PHI1R  ,THET1R,G     ,RI     ,H     ,
0955          .               PHIR  ,THETR ,F10    ,F10B   ,AP    ,IHR    ,MIN   ,
0956          .               NMORE ,DX    ,HL     ,VL     ,DZ    ,B      ,EPS   ,
0957          .               IOPP  ,LOOK  ,IET    ,GLAT   ,RP1S  ,RD1S   ,RT1S  ,
0958          .               RU1S  ,RV1S  ,SP1S   ,SD1S   ,ST1S  ,SU1S   ,SV1S  ,
0959          .               UDS1  ,VDS1  ,UDL1   ,VDL1   ,UDS2  ,VDS2   ,UDL2  ,
0960          .               VDL2  ,REARTH
0961          COMMON /CHIC  / LA(4,4),NB(2),IWSYM,UCOEF(14,9),VCOEF(14,9)
0962          COMMON /WINCOM/ DUMSTF(17),UPRE,VPRE,DUPRE,DVPRE
0963    C
0964          DATA PI  /3.141593 /
0965          DATA FAC /0.01745329/
0966    C
0967          LOOK = 0
0968          H    = 0.0
0969          NSAME = 0
0970    C
0971    C.... FIRST READ DEFINES INITIAL HEIGHT (KM), INITIAL LATITUDE (DEG)
0972    C     INITIAL LONGITUDE (DEG), F10.7, MEAN F10.7, AP, MONTH, DAY,
```

```
0973   C      YEAR (TOTAL YEAR - 1900), GREENWICH HOUR, MINUTES, SECONDS,
0974   C      LATITUDE INCREMENT (DEG), LONGITUDE INCREMENT (DEG),
0975   C      HEIGHT DECREASE (KM), MAXIMUM NUMBER OF POSITIONS (EXCLUDING
0976   C      INITIAL POSITION) TO BE COMPUTED, TIME INCREMENT BETWEEN
0977   C      POSITIONS, TRAJECTORY OPTION, OUTPUT OPTION, MINIMUM GEOSTROPHIC
0978   C      LATITUDE.
0979   C
0980   C... SET INITIAL CONDITIONS FOR INITIALIZATION PROCESS
0981          H1    = ALTINT/1000.
0982          PHI1  = XLATIN
0983          THET1 = XLONIN
0984          MN    = ICLIMT
0985   C... READ FIXED INPUT DATA
0986          OPEN(UNIT=55,FILE='FIXED_INPUT.',STATUS='OLD',READONLY)
0987          READ(55,*,END=90) F10    ,F10B  ,AP      ,
0988          .                 IDA    ,IYR   ,IHRO  ,MINO  ,ISECO ,
0989          .                 DPHI   ,DTHET ,DH     ,NMAX  ,INCT  ,IOPT  ,
0990          .                 IOPP   ,GLAT
0991   C
0992          GLAT = ABS(GLAT)
0993          IF (GLAT.LT. 5.) GLAT =  5.
0994          IF (GLAT.GE.18.) GLAT = 17.999
0995          GLATF=GLAT*FAC
0996   C
0997   C.... INITIALIZE DATA ARRAYS.
0998   C
0999          CALL SETUP
1000   C       CALL GRIDIN
1001   C
1002          CLOSE(55)
1003          RETURN
1004   C
1005     90 CONTINUE
1006   C
1007          WRITE(6,555)
1008    555 FORMAT('  GRAMIN PROBLEM')
1009          STOP
1010          END
1011   C
1012          SUBROUTINE GRAMRT (FIRST)
1013   C
1014   C.... THIS ROUTINE IS THE EXECUTIVE FOR THE REAL-TIME GRAM PROGRAM.
1015   C
1016   C.... WRITTEN   26 JAN 89   L SCHILLING   NASA/ADFRF.
1017   C
1018   C.... GRAM INPUTS:
1019   C                         GRAM            DRYDEN SIM
1020   C                         --------------  --------------
1021   C      TIME              SEC             SEC
1022   C      ALTITUDE          KILOMETERS      FEET
1023   C      LATITUDE          DEG, +NORTH     DEG, +NORTH
1024   C      LONGITUDE         DEG, +WEST      DEG, +EAST
1025   C
1026   C.... GRAM OUTPUTS:
1027   C
1028   C             UNPERTURBED (MONTHLY MEAN)
1029   C             ==========================================
1030   C                         GRAM            DRYDEN SIM
1031   C                         --------------  --------------
1032   C      PRESSURE          NEWTONS/METER**2  POUNDS/FT**2
1033   C      DENSITY           KILOGRAMS/METER**3 SLUGS/FT**3
1034   C      TEMPERATURE       DEGREES KELVIN    DEGREES RANKINE
1035   C      GEOSTROPHIC WIND  METERS/SEC        FEET/SEC
1036   C
```

230

```
1037    C                          MEAN PLUS PERTURBATIONS
1038    C            --------------------------------------------------
1039    C                               GRAM              DRYDEN SIM
1040    C                               --------------    --------------
1041    C       PRESSURE               NEWTONS/METER**2   POUNDS/FT**2
1042    C       DENSITY                KILOGRAMS/METER**3 SLUGS/FT**3
1043    C       TEMPERATURE            DEGREES KELVIN     DEGREES RANKINE
1044    C       TOTAL WIND             METERS/SEC         FEET/SEC
1045    C
1046    C       THERMAL WIND SHEAR  METERS/SEC/KILOMETER  FEET/SEC/FOOT
1047    C
1048            LOGICAL FIRST
1049    C
1050    C.... GRAM PROGRAM COMMON BLOCKS.
1051    C
1052            COMMON /C4     /
1053       .                    GLAT(16),GLON(16),NG     ,P4D(16,26),D4D(16,26),
1054       .                    T4D(16,26),SP4(16,26),SD4(16,26),ST4(16,26),
1055       .                    THET1 ,THET  ,HS
1056            COMMON /CHIC  / LA(4,4),NB(2),IWSYM,UCOEF(14,9),VCOEF(14,9)
1057            COMMON /COMJAC/ LAT_RADJ ,XLONG ,SDA    ,SHA    ,DY     ,R88    ,TE    ,
1058       .                    EM
1059            COMMON /COMPER/ SPH    ,SDH    ,STH    ,PRH    ,DRH    ,TRH    ,URH    ,
1060       .                    VRH    ,SUP    ,SVH    ,CP     ,PRHS   ,DRHS   ,TRHS   ,
1061       .                    URHS   ,VRHS   ,PRHL   ,DRHL   ,TRHL   ,URHL   ,VRHL   ,
1062       .                    SPHS   ,SDHS   ,STHS   ,SUHS   ,SVHS   ,SPHL   ,SDHL   ,
1063       .                    STHL   ,SUHL   ,SVHL
1064            COMMON /IOTEMP/ IOTEM1,IOTEM2,IUG    ,IUN    ,DD     ,XMJD   ,PHI1   ,
1065       .                    PHI    ,NSAME  ,RP1    ,RD1    ,RT1    ,SP1    ,SD1    ,
1066       .                    ST1    ,RU1    ,RV1    ,SU1    ,SV1    ,MN     ,IDA    ,
1067       .                    IYR    ,H1     ,PHI1R  ,THET1R,G      ,RI     ,H      ,
1068       .                    PHIR   ,THETR  ,F10    ,F10B   ,AP     ,IHR    ,MIN    ,
1069       .                    NMORE  ,DX     ,HL     ,VL     ,DZ     ,B      ,EPS    ,
1070       .                    IOPP   ,LOOK   ,IET    ,GLATX  ,RP1S   ,RD1S   ,RT1S   ,
1071       .                    RU1S   ,RV1S   ,SP1S   ,SD1S   ,ST1S   ,SU1S   ,SV1S   ,
1072       .                    UDS1   ,VDS1   ,UDL1   ,VDL1   ,UDS2   ,VDS2   ,UDL2   ,
1073       .                    VDL2   ,REARTH
1074            COMMON /IPRTP / IPRT
1075            COMMON /PDTCOM/
1076       .                    IU4    ,MONTH  ,IOPR   ,PG(18,19),TG(18,19),
1077       .                    DG(18,19),PSP(8,10,12),DSP(8,10,12),TSP(8,10,12),
1078       .                    PAQ(17,5),DAQ(17,5),TAQ(17,5),PDQ(17,5),DDQ(17,5),
1079       .                    TDQ(17,5),PR(20,10),DR(20,10),TR(20,10),UAQ(17,5),
1080       .                    VAQ(17,5),UDQ(17,5),VDQ(17,5),UR(25,10),VR(25,10),
1081       .                    PQ     ,DQ     ,TQ     ,UQ     ,VQ     ,PQA    ,DQA    ,
1082       .                    TQA    ,UA     ,VA     ,IOPQ   ,PLP(25,10),DLP(25,10),
1083       .                    TLP(25,10),ULP(25,10),VLP(25,10),UDL(25,10),
1084       .                    VDL(25,10),UDS(25,10),VDS(25,10)
1085            COMMON /WINCOM/ DH     ,FCORY  ,DX5    ,DY5    ,DPX    ,DPY    ,DPXX   ,
1086       .                    DPXY   ,DPYY   ,UGH    ,VGH    ,TH     ,DTX    ,DTY    ,
1087       .                    DUH    ,DVH    ,PH     ,UPRE   ,VPRE   ,DUPRE  ,DVPRE
1088    C
1089    C.... COMMON BLOCKS ADDED IN MODIFYING GRAM AND INTERFACING WITH SIM.
1090    C
1091            COMMON /GRAMOT/ PGH    ,DGH    ,TGH    ,UH     ,VH     ,PS     ,DS     ,
1092       .                    TS     ,PGHP   ,DGHP   ,TGHP   ,PHP    ,DHP    ,THP    .
1093       .                    PSH    ,DSH    ,TSH    ,WGH
1094            LOGICAL         GPMATM,G76ATM,GATMF  ,GRMWND,GWINDP
1095            COMMON /GRMDAT/ GPMATM,G76ATM,GATMF  ,G_MWND,GWINDP,CS76   ,CSU    ,
1096       .                    CSP    ,TMPP76,TMPRU  ,TMPPF  ,PA76   ,PAU    ,PAF    ,
1097       .                    RHO76  ,RHOU   ,RHOF   ,UWINDU,UWINDP,VWINDU,VWINDP,
1098       .                    USHEAR,VSHEAR
1099            COMMON /NASPGM/ PDAT(5720) ,DDAT(5720) ,TDAT(5720) ,
1100       .                    SPDAT(5720),SDDAT(5720),STDAT(5720)
```

231

```
1101          LOGICAL           WIND
1102          COMMON /WINDAT/ WIND   ,XWIND ,YWIND ,NUMWND,ALTW(32),VELW(32),
1103        .                 HDGW(32),XWA(32),YWA(32)
1104   C
1105   C.... DRYDEN SIM COMMON BLOCKS.
1106   C
1107          COMMON /ALTFUN/ ASIM   ,RHO    ,GSIM   ,PA      ,TMPR
1108          REAL*8 FSIM, T, XLAT, XLNG
1109          COMMON /DRVOUT/ FSIM(13),DFSIM(13)
1110          COMMON /DRVOT2/ ALP    ,ALPDOT,BTA    ,BTADOT,HSIM   ,HDOT   ,
1111        .                 V      ,VDOT  ,X      ,XDOT  ,Y      ,YDOT ,VI
1112          LOGICAL           OP,RST,HLD,RT,ATRM,ICEN,MDAT,HAZ
1113          COMMON /RTCDAT/ OP,RST,HLD,RT,ATRM,ICEN,MDAT,HAZ
1114          EQUIVALENCE (FSIM( 1),T      ),(FSIM(12),XLAT),(FSIM(13),XLNG)
1115   C
1116          DATA R2D /57.29578/
1117   CCC      DATA FIRST /.TRUE./
1118   C
1119   C.... FIRST TIME IN REAL-TIME LOOP, INITIALIZE RANDOM NUMBER
1120   C      GENERATOR.  REQUIRED BECAUSE 'SETUP', WHICH INITIALIZES
1121   C      'RAND' IN THE STANDARD GRAM PROGRAM, IS NOT IN THE REAL-
1122   C      TIME LOAD MODULE IN THE DRYDEN SIM.  LJS.
1123   C
1124          IF(FIRST) THEN
1125             RDUM=RAND(1)
1126             RDUM=RAND(0)
1127             RDUM=RAND(0)
1128             FIRST=.FALSE.
1129             RST=.TRUE.
1130             OP=.FALSE.
1131             HLD=.FALSE.
1132          ELSE
1133             RST=.FALSE.
1134             OP=.TRUE.
1135             HLD=.FALSE.
1136          ENDIF
1137   C
1138          GLATF = GLATX / R2D
1139   C
1140   C.... SCALE INPUTS FROM SIM.
1141   C
1142          IET  = T
1143          H    = HSIM/3280.84
1144          PHI  = XLAT*R2D
1145          THET = XLNG*R2D
1146          IF(THET.LT.0.0) THET = THET+360.0
1147   C
1148   C.... IF HOLD MODE, RETURN.
1149   C
1150          IF(HLD) RETURN
1151   C
1152   C.... IF RESET MODE, EXECUTE GRAM FIRST PASS CODE FROM GRAM MAIN
1153   C      PROGRAM.
1154   C
1155          IF(.NOT.RST) GO TO 200
1156   C
1157          NT   = 1
1158          PHIR =PHI /R2D
1159          THETR=THET/R2D
1160   C
1161          PHI1R  = PHIP
1162          THET1R = THETP
1163   C
1164   C.... A=EQUATORIAL EARTH RADIUS, B = POLAR EARTH RADIUS
```

```
1165    C       EPS= EARTH ECCENTRICITY
1166    C
1167            A  = 6378.160
1168            B  = 6356.7747
1169            EPS=(1.-(B*B)/(A*A))
1170    C
1171    C.... COMPUTES RADIUS TO HEIGHT H, AND GRAVITY AT HEIGHT AND
1172    C       LATITUDE PHIR
1173    C
1174            CALL RIG
1175            ISEC=ISECO+IET
1176            ISEC=MOD(ISEC,60)
1177            MIN = MINO + IET/60
1178            IHR = IHRO + MIN / 60
1179            MIN = MOD(MIN,60)
1180    C
1181    C.... COMPUTES P,D,T,U,V AT FIRST POSITION AFTER INITILL POSITION
1182    C
1183            IF(H.LE.30.) LOOK=1
1184            IF(ABS(PHIR).GT.GLATF) GO TO 195
1185            IF(H.GE.25.0 .AND. H.LE.90.0) GO TO 195
1186            PHI1S=PHI1R
1187            PHIS=PHIR
1188            DPHIS=(PHIR+GLATF)/(2.*GLATF)
1189            PHIR=GLATF
1190            PHI1R=PHIR+PHI1S-PHIS
1191    C
1192            CALL SCIMOD(0)
1193    C
1194            UP2=UPRE
1195            VP2=VPRE
1196            DUP2=DUPRE
1197            DVP2=DVPRE
1198            PHIR=-GLATF
1199    C
1200            CALL SCIMOD(0)
1201    C
1202            UP1=UPRE
1203            VP1=VPRE
1204            DUP1=DUPRE
1205            DVP1=DVPRE
1206            UPRE=UP1+(UP2-UP1)*DPHIS
1207            VPRE=VP1+(VP2-VP1)*DPHIS
1208            DUPRE=DUP1+(DUP2-DUP1)*DPHIS
1209            DVPRE=DVP1+(DVP2-DVP1)*DPHIS
1210            PHIR=PHIS
1211            PHI1R=PHI1S
1212    C
1213      195 CALL SCIMOD(1)
1214    C
1215      200 CONTINUE
1216    C
1217    C.... IF OPERATE MODE, CYCLE GRAM PROGRAM.  CODE FROM GRAM 'MAIN'.
1218    C
1219            IF(.NOT.OP) GO TO 300
1220    C
1221            NT = NT + 1
1222            PHIR =PHI /R2D
1223            THETR=THET/R2D
1224            MIN=MINO+IET/60
1225            ISEC=IET
1226            ISEC=MOD(ISEC,60)
1227            IHR=IHRO+MIN/60
1228            MIN=MOD(MIN,60)
```

```
1229    C
1230    C.... COMPUTE RADIUS AND GRAVITY AT NEW POSITION
1231    C
1232          CALL RIG
1233    C
1234    C.... COMPUTE P,D,T,U,V, AT NEW POSITION
1235    C
1236          IF(ABS(PHIR).GT.GLATF) GO TO 80
1237          IF(H.GE.25.0 .AND. H.LE.90.0) GO TO 80
1238          PHI1S=PHI1R
1239          PHIS=PHIR
1240          DPHIS=(PHIR+GLATF)/(2.*GLATF)
1241          PHIR=GLATF
1242          PHI1R=PHIR+PHI1S-PHIS
1243    C
1244          CALL SCIMOD(0)
1245    C
1246          UP2=UPRE
1247          VP2=VPRE
1248          DUP2=DUPRE
1249          DVP2=DVPRE
1250          PHIR=-GLATF
1251    C
1252          CALL SCIMOD(0)
1253    C
1254          UP1=UPRE
1255          VP1=VPRE
1256          DUP1=DUPRE
1257          DVP1=DVPRE
1258          UPRE=UP1+(UP2-UP1)*DPHIS
1259          VPRE=VP1+(VP2-VP1)*DPHIS
1260          DUPRE=DUP1+(DUP2-DUP1)*DPHIS
1261          DVPRE=DVP1+(DVP2-DVP1)*DPHIS
1262          PHIR=PHIS
1263          PHI1R=PHI1S
1264    C
1265       80 CALL SCIMOD(1)
1266    C
1267      300 CONTINUE
1268    C
1269    C.... SCALE GRAM OUTPUTS FOR SIMULATION USE.
1270    C
1271    C.... COMPUTE SPEED OF SOUND IN M/SEC.
1272    C
1273          CS76MS=SQRT(401.8743*TS )
1274          CSUNPR=SQRT(401.8743*TGH)
1275          CSPERT=SQRT(401.8743*TH )
1276    C
1277    C.... CONVERT FROM SI TO ENGLISH UNITS.  '76' SUFFIX USED FOR THE 76
1278    C     STANDARD ATMOSPHERE VALUE.  'U' SUFFIX USE FOR THE GRAM
1279    C     UNPERTURBED (MONTHLY MEAN) VALUE.  'P' SUFFIX USED FOR THE GRAM
1280    C     MEAN PLUS PERTURBATIONS.
1281    C
1282          CS76  =CS76MS/0.3048            !SPEED OF SOUND IN FT/SEC
1283          CSU   =CSUNPR/0.3048
1284          CSP   =CSPERT/0.3048
1285    C
1286          TMPR76=TS *1.8                  !TEMPERATURE IN DEG RANKINE
1287          TMPRU =TGH*1.8
1288          TMPRP =TH *1.8
1289    C
1290          PA76  =PS *0.02088543           !PRESSURE IN LBS/FT**2
1291          PAU   =PGH*0.02088543
1292          PAP   =PH *0.02088543
```

234

```
1293    C
1294          RHO76 =DS *0.001940319          !DENSITY IN SLUGS/FT**3
1295          RHOU  =DGH*0.001940319
1296          RHOP  =DH *0.001940319
1297    C
1298          UWINDU=VGH/0.3048               !GEOSTROPHIC & TOTAL WINDS FT/SEC
1299          UWINDP=VH /0.3048               !U,V COMPONENTS INTERCHANGE
1300          VWINDU=UGH/0.3048               !BETWEEN GRAM AND SIM AXES.
1301          VWINDP=UH /0.3048
1302    C
1303          USHEAR=DVH/1000.0               !WIND SHEAR (FT/SEC)/FT
1304          VSHEAR=DUH/1000.0
1305    C
1306    C.... SELECT OUTPUT TO SIM BASED ON USER SELECTION.  IT IS POSSIBLE
1307    C     FOR THE USER TO SELECT GRAM ATMOSPHERE AND GRAM WINDS
1308    C     INDEPENDENTLY.
1309    C
1310          IF(.NOT.GRMATM) GO TO 50
1311    C
1312          IF(G76ATM) THEN             ! GRAM 76 REFERENCE SELECTED
1313             ASIM = CS76
1314             RHO  = RHO76
1315             PA   = PA76
1316             TMPR = TMPR76
1317          ELSE                        ! GRAM CALCULATED ATMOSPHERE SELECTED
1318             IF(GATMP ) THEN          ! GRAM MONTHLY MEAN + PERTURBATIONS
1319                ASIM = CSP
1320                RHO  = RHOP
1321                PA   = PAP
1322                TMPR = TMPRP
1323             ELSE                     ! GRAM MONTHLY MEAN WITHOUT PERT.
1324                ASIM = CSU
1325                RHO  = RHOU
1326                PA   = PAU
1327                TMPR = TMPRU
1328             ENDIF
1329          ENDIF
1330    C
1331       50 CONTINUE
1332    C
1333    C.... TEST FOR USER SELECTION OF WINDS.
1334    C
1335          IF(.NOT.GRMWND) GO TO 60
1336    C
1337          IF(WIND) THEN               ! GRAM WINDS SELECTED, WIND ON
1338             IF(GWINDP) THEN          ! MEAN + PERTURBATIONS SELECTED
1339                XWIND = UWINDP
1340                YWIND = VWINDP
1341             ELSE                     ! MONTHLY MEAN WITHOUT PERT.
1342                XWIND = UWINDU
1343                YWIND = VWINDU
1344             ENDIF
1345          ELSE                        ! GRAM WINDS SELECTED, BUT WINDS OFF.
1346             XWIND = 0.0
1347             YWIND = 0.0
1348          ENDIF
1349    C
1350       60 CONTINUE
1351    C
1352          RETURN
1353          END
1354    C
1355          SUBROUTINE GRIDIN
1356    C
```

```
1357   C.... THIS ROUTINE READS A BINARY DATA FILE CONSISTING OF PRESSURE,
1358   C      DENSITY, TEMPERATURE, PRESSURE VARIANCE, DENSITY VARIANCE, AND
1359   C      TEMPERATURE VARIANCE FOR LATITUDES 20-65 AND LONGITUDES 35-140
1360   C      WEST (CONTINENTAL US +).  THE DATA TABLE IS USED TO SUPPLY THE
1361   C      REAL-TIME GRAM PROGRAM WITH 4D GRID DATA WHEN WITHIN THE REGION
1362   C      INDICATED ABOVE (CONUS INCLUDING A SORROUNDING AREA).  THE
1363   C      DATA IS USED BY ROUTINE 'USGRID' TO BUILD A GRID.  'USGRID'
1364   C      REPLACES 'GEN4D', 'ADJUST', 'GRID4D', 'INTRP4', 'SELEC4', AND
1365   C      'SORT4'.
1366   C
1367   C.... THIS ROUTINE REQUIRES THE INCLUSION OF THE EXTENDED REGION.  THE
1368   C      REGION IS CREATED ON THE SYSTEM VOLUME WITH THE FOLLOWING
1369   C      VOLUME MANAGER COMMAND:
1370   C         CREATE COMMON NASPGM FIRST=380 PROT=68 ACCESS=OT(R W)
1371   C
1372   C.... ROUTINE I/O:   LUN 25 IS INPUT FILE
1373   C                    'OT'  IS TERMINAL OUTPUT
1374   C
1375   C.... WRITTEN   23 JAN 89   L SCHILLING   NASA/ADFRF.
1376   C
1377         CHARACTER*12 FILNAM
1378         COMMON /PDTCOM/ IT,MONTH,DUMMY1(8118)
1379         COMMON /NASPGM/ PDAT(5720) ,DDAT(5720) ,TDAT(5720) ,
1380        .                  SPDAT(5720),SDDAT(5720),STDAT(5720)
1381   C
1382   C.... OPEN BINARY FILE CONTAINING US 4D GRID DATA FOR MONTH OF
1383   C      INTEREST.
1384   C
1385         WRITE(FILNAM,777) MONTH
1386     777 FORMAT('NASPGRID',I2,'.B')
1387         IF(FILNAM(9:9).EQ.' ') FILNAM(9:9)='0'
1388   C
1389         OPEN(25,FILE=FILNAM,STATUS='OLD',FORM='UNFORMATTED',
1390        .     ACCESS='SEQUENTIAL',ERR=99,IOSTAT=IOS)
1391   C
1392   C.... READ IN BINARY DATA FILE.
1393   C
1394         REWIND(25)
1395         READ(25) PDAT,TDAT,DDAT,SPDAT,STDAT,SDDAT
1396   C
1397         CLOSE(25)
1398   C
1399         RETURN
1400   C
1401   C.... ERROR CONDITION.
1402   C
1403      99 CONTINUE
1404   C
1405         WRITE(6,231) FILNAM,IOS
1406     231 FORMAT('  OPEN ERROR ON ',A12,'  STATUS=',I3)
1407   C
1408         STOP
1409         END
1410   C
1411         SUBROUTINE GTEPF(IH,PHI,P,D,T,PG,DG,TG,DPY,DTY,DP2Y)
1412   C
1413   C.....INTERPOLATES GROVES DATA TO HEIGHT IH AND LATITUDE PHI
1414   C
1415   C.... DECLARE APPROPRIATE ARGUMENTS TO BE IN GOULD EXTENDED.  IF A
1416   C      NON-EXTENDED PARAMETER IS DECLARED EXTENDED, THE INTERFACE WILL
1417   C      WORK, ALBEIT WITH A SLIGHT EXECUTION OVERHEAD PENALTY.  LJS.
1418   C
1419   C
1420         DIMENSION PG(18,19),TG(18,19),DG(18,19)
```

```
1421   C      HEIGHT INDEX
1422          I =(IH - 20)/5
1423   C      LOWER LATITUDE INDEX
1424          J = INT((PHI + 100.)/10.)
1425          IF (J.LT.1) J = 1
1426          IF (J.GT.18) J = 18
1427   C      UPPER LATITUDE INDEX
1428          JP = J + 1
1429   C.....CHECK FOR DENSITY OR TEMPERATURE LEQ 0
1430          CHK = DG(I,J) * TG(I,J) * DG(I,JP) * TG(I,JP)
1431          IF (CHK) 10,10,20
1432     10   P = PG(I,J)
1433          D = DG(I,J)
1434          T = TG(I,J)
1435          GO TO 30
1436   C.....LATITUDE DEVIATION FROM GROVES ARRAY POSITION
1437     20   PHIF = (PHI + 100. - 10.*J)/10.
1438          TL= TG(I,J) + (TG(I,JP ) - TG(I,J))*PHIF
1439   C       LATITUDE INTERPOLATION
1440          DL= DG(I,J) + (DG(I,JP ) - DG(I,J)) *PHIF
1441          R1 = PG(I,J)/(DG(I,J)*TG(I,J))
1442          R2 = PG(I,JP)/(DG(I,JP)*TG(I,JP))
1443   C      INTERPOLATED GAS CONSTANT
1444          R = R1 + (R2 - R1)*PHIF
1445   C      PRESSURE COMPUTED FROM INTERPOLATED GAS CONSTANT
1446          P =DL*R*TL
1447          D = DL
1448          T = TL
1449   C      DP/DY FOR GEOSTOPHIC WINDS
1450     30   DPY = (PG(I,JP) - PG(I,J)) * 0.5
1451   C      DT/DY FOR THERMAL WINDS
1452          DTY = (TG(I,JP) - TG(I,J)) * 0.5
1453          JM = J - 1
1454          IF (JM.LT.1) JM = JP
1455          DP2Y = (PG(I,JP) - PG(I,JM ))*0.5
1456          IF (ABS(PHI)-90.) 50,40,40
1457     40   DPY = 0.
1458          DTY = 0.
1459          DP2Y = 0.
1460     50   CONTINUE
1461          RETURN
1462          END
1463   C
1464          SUBROUTINE INTLL(F,IA,IB,IC,ID,FLL,GLAT,GLON,CLAT,CLON,IH)
1465   C
1466   C.....INTERPOLATES FUNCTION (ARRAY) F FROM VALUES OF GLAT AND GLON AT
1467   C       INDEX VALUES IA, IB, IC, ID TO OUTPUT VALUE FLL AT HEIGHT IH
1468   C       AND POSITION CLAT, CLON
1469   C
1470          DIMENSION F(16,26),GLAT(16),GLON(16)
1471   C.....NORMALIZES LONGITUDE DISPLACEMENT
1472          IF(F(IA,IH)*F(IB,IH)*F(IC,IH)*F(ID,IH)) 20,10,20
1473     10 FLL=0.
1474          RETURN
1475     20 X=(CLON-GLON(IB))/(GLON(IA)-GLON(IB))
1476   C.....NORMALIZES LATITUDE DISPLACEMENT
1477          Y=(CLAT-GLAT(IA))/(GLAT(IC)-GLAT(IA))
1478   C.....TWO DIMENSIONAL INTERPOLATION
1479          FLL=F(IB,IH)+(F(ID,IH)-F(IB,IH))*Y+(F(IA,IH)-F(IB,IH))*X
1480        1  +(F(IC,IH)-F(IA,IH)-F(ID,IH)+F(IB,IH))*X*Y
1481          RETURN
1482          END
1483          SUBROUTINE INTRUV(UR,VR,H,PHI,SUH,SVH)
1484   C
```

```
1485    C.....FINDS RANDOM WIND STANDARD DEVIATION AT HEIGHT H (KM), LATITUDE
1486    C         PHI (DEGREES), FROM UR AND VR ARRAYS
1487    C
1488          DIMENSION UR(25,10),VR(25,10)
1489    C.....I - LOWER HEIGHT INDEX
1490          IF (H.LT.95.) I = 1 + INT(H) / 5
1491          IF (H.GE.95.) I=19+(INT(H)-80)/20
1492          IF (I.GT.25) I = 25
1493    C     UPPER HEIGHT INDEX
1494          IP=I+1
1495          IF (IP.GT.25) IP=25
1496    C     LOWER LATITUDE INDEX
1497          J=INT(PHI+110.)/20
1498    C      UPPER LATITUDE INDEX
1499          JP=J+1
1500          IF (JP.GT.10) JP=10
1501    C.....PHI1 - LOWER LATITUDE FOR UR AND VR ARRAY VALUES
1502          PHI1=-110.+20.*J
1503    C.....PHI2 - UPPER LATITUDE FOR UR AND VR ARRAY VALUES
1504          PHI2=-110.+20.*JP
1505          IF (I.GT.19) GO TO 10
1506    C      LOWER HEIGHT FOR UR AND VR ARRAY VALUES
1507          Z1=5.*(I-1)
1508          GO TO 20
1509    10     Z1=20.*(I-15)
1510    20     IF (IP.GT.19) GO TO 30
1511    C      UPPER HEIGHT FOR UR AND VR ARRAY VALUES
1512          Z2=5.*(IP-1)
1513          GO TO 40
1514      30 Z2 = 20. * (IP - 15)
1515    C     INTERPOLATE ON LATITUDE AT LOWER HEIGHT
1516      40 CALL INTERW(UR(I,J),VR(I,J),PHI1,UR(I,JP),VR(I,JP),PHI2,U1,V1,
1517        $ PHI)
1518    C      INTERPOLATE ON LATITUDE AT UPPER HEIGHT
1519          CALL INTERW(UR(IP,J),VR(IP,J),PHI1,UR(IP,JP),VR(IP,JP),PHI2,U2,
1520        $ V2,PHI)
1521    C      INTERPOLATE ON HEIGHT
1522          CALL INTERW(U1,V1,Z1,U2,V2,Z2,SUH,SVH,H)
1523          RETURN
1524          END
1525          SUBROUTINE INTERW(U1,V1,Z1,U2,V2,Z2,U,V,Z)
1526    C
1527          IF ( Z1 - Z2 ) 20,10,20
1528      10 U = U1
1529    C     SETS U,V = U1,V1 IF Z1 = Z2
1530          V = V1
1531          RETURN
1532      20 A = (Z-Z1)/(Z2-Z1)
1533          U = U1 + (U2-U1) * A
1534          V = V1 + (V2-V1) * A
1535    C.....LINEAR INTERPOLATION BETWEEN U1,V1 AT HEIGHT Z1 AND U2,V2 AT
1536    C         HEIGHT Z2. OUTPUT IS U,V AT HEIGHT Z
1537          RETURN
1538          END
1539          SUBROUTINE INTERZ(P1,D1,T1,Z1,P2,D2,T2,Z2,P,D,T,Z)
1540    C
1541       5 IF (Z1 - Z2)  20,10,20
1542      10   P = P1
1543          D = D1
1544    C     SETS P, D, T = P1,D1,T1, IF Z1 = Z2
1545          T =T1
1546          RETURN
1547      20   A = (Z - Z1)/ (Z2 - Z1)
1548          T = T1 + (T2 - T1) * A
```

```
1549            D = D1 + (D2 - D1) * A
1550            P = P1 + (P2 - P1) * A
1551    C.....LINEAR INTERPOLATION BETWEEN P1,D1,T1 AT HEIGHT Z1 AND P2,D2,T2
1552    C         AT HEIGHT Z2 TO OUTPUT VALUES OF P,D,T AT HEIGHT Z
1553            RETURN
1554            END
1555            SUBROUTINE INTER2(P1,D1,T1,Z1,P2,D2,T2,Z2,P,D,T,Z)
1556    C.....INTERPOLATES BETWEEN P1,D1,T1 AT HEIGHT Z1 AND P2,D2,T2 AT
1557    C         HEIGHT Z2 TO OUTPUT VALUES OF P,D,T AT HEIGHT Z
1558    C.....CHECKS FOR T1,D1,T2,D2 PRODUCT = 0, FOR GAS CONSTANT INTERPOLATION
1559            CHK=T1*D1*T2*D2
1560            IF (CHK) 10,10,5
1561          5 IF (Z1 - Z2)  20,10,20
1562     10     P = P1
1563            D = D1
1564    C         SETS P,D,T = P1,D1,T1 IF Z1=Z2
1565            T = T1
1566            RETURN
1567     20 IF(P1*D1*T1*P2*D2*T2.LE.0.)GO TO 30
1568            IF(D2*D1.LE.0.0)GO TO 30
1569            A=ALOG(D2/D1)/(Z2-Z1)
1570    C        LINEAR INTERPOLATION ON LOG D
1571            DZ= D1*EXP(A*(Z - Z1))
1572            A=(Z-Z1)/(Z2-Z1)
1573    C        LINEAR INTERPOLATION ON T
1574            TZ= T1 + A*(T2-T1)
1575            R1=P1/(D1*T1)
1576            R2=P2/(D2*T2)
1577    C        LINEAR INTERPOLATION ON GAS CONSTANT R
1578            R=(R2-R1)*A+R1
1579    C         PRESSURE FROM PERFECT GAS LAW
1580            P = DZ * R * TZ
1581            D = DZ
1582            T = TZ
1583            RETURN
1584     30 P=0.
1585            D=0.
1586            T=0.
1587            RETURN
1588            END
1589            SUBROUTINE INTER4 (            CLAT, CLON, IZ,      P, D, T,
1590          $ P4, D4, T4, DPX, DPY, DTX, DTY,DPXX,DPYY,DPXY)
1591    C
1592            COMMON/IOTEMP/IOTEM1,IOTEM2,IUG, IUN ,DD,XMJD,PHI1,PHI,
1593           $NSAME,DUMMY2(56)
1594    C.....INTERPOLATES BETWEEN 4D ARRAYS P(I,IH),D(I,IH),T(I,IH) AT GRID
1595    C        LOCATIONS LATITUDE GLAT(I) LONGITUDE GLON(I).
1596    C        CLAT,CLON = CURRENT LATITUDE,LONGITUDE
1597    C        IZ - HEIGHT                  NG = NUMBER OF 4D GRID POSITIONS
1598    C        OUTPUT = P4,D4,T4, AND DERIVATIVES DPX,DPY,DTX,DTY
1599            COMMON /C4     / GLAT(16),GLON(16),NG,DUMMY(2499)
1600            COMMON/CHIC/LA(4,4),NB(2),IWSYM,UCOEF(14,9),VCOEF(14,9)
1601            DIMENSION                  P(16,26),D(16,26),T(16,26),LAX(16)
1602            DATA IBLK/1H /,IAST/1H*/
1603            IWSYM = IBLK
1604            ICHK = 0
1605    C       HEIGHT INDEX = HEIGHT + 1
1606            IH = IZ + 1
1607          5 IF (ICHK.GT.1) GO TO 220
1608            IF (NG.GT.9) GO TO 100
1609    C       NG = 9 MEANS POLAR GRID
1610            DO 10 I=10,16,1
1611            P(I,IH) = P(9,IH)
1612            D(I,IH) = D(9,IH)
```

239

```
1613              T(I,IH) = T(9,IH)
1614              GLAT(I) = GLAT(9)
1615    C         I=10-16 ALL AT 90 DEG
1616         10 GLON(I) = GLON(I-8)
1617    C         LOWER RIGHT INTERPOLATION INDEX
1618              IB = INT(CLON/45) + 1
1619    C         LOWER LEFT INTERPOLATION INDEX
1620              IA =IB+1
1621              IF (IA.GT.8) IA = IA-8
1622    C         POSITION OUTSIDE POLAR GRID
1623              IF (ABS(CLAT).LT.75.) GO TO 20
1624    C         UPPER LEFT INTERPOLATION INDEX
1625              IC = IA +8
1626    C         UPPER RIGHT INTERPOLATION INDEX
1627              ID = IB + 8
1628              GO TO 300
1629         20  IF(NSAME.EQ.1) NSAME=2
1630              CALL GEN4D
1631    C::_
1632    C         CALL USGRID
1633    C::^
1634              ICHK = ICHK + 1
1635              GO TO 5
1636        100 XLON = CLON
1637              DO 105 I = 1,4
1638              DO 105 J = 1,4
1639              I16 = 4*(I-1) + J
1640              LAX(I16) = LA(I,J)
1641        105   CONTINUE
1642              IF (XLON-GLON(1).GT.180)XLON=CLON-360.
1643    C.....CHECKS FOR POSITION WITHIN 16 POINT GRID 110=GOOD.  200=POSITION
1644    C          OUTSIDE GRID.
1645              IF (CLAT.GE.GLAT(1) .AND. CLAT.LT.GLAT(16)  .AND. XLON.LE.GLON(1)
1646           $   .AND.XLON.GT.GLON(16)) GO TO 110
1647              GO TO 200
1648        110 NDL=5
1649              IF(ABS(CLAT).LT.18)     =12
1650              IA = 1 + INT((GLON(1  - XLON) / 5)
1651    C.....IA = LOWER LEFT (REFERENCE) INTERPOLATION INDEX
1652              IA = IA + 4 * INT((CLAT - GLAT(1)) / NDL)
1653    C         LOWER RIGHT INTERPOLATION INDEX
1654              IB = IA + 1
1655    C         UPPER LEFT INTERPOLATION INDEX
1656              IC = IA + 4
1657    C         UPPER RIGHT INTERPOLATION INDEX
1658              ID = IA + 5
1659              GO TO 300
1660        200  IF(NSAME.EQ.1)NSAME=2
1661              CALL GEN4D
1662    C::_
1663    C         CALL USGRID
1664    C::^
1665              ICHK = ICHK + 1
1666              GO TO 5
1667        220 CONTINUE
1668    C:: 220 WRITE(6,250)
1669    C:: 250 FORMAT(1H , 'UNABLE TO GENERATE 4-D GRID.  TOO MANY ',
1670    C::      &'RETRIES IN INTER4')
1671              P4=0.
1672              D4=0.
1673              T4=0.
1674              RETURN
1675    C.....INTERPOLATION FOR POSITION INSIDE 16 POINT GRID OR POLAR GRID
1676        300 CALL INTLL(P,IA,IB,IC,ID,P4,GLAT,GLON,CLAT,XLON,IH)
```

240

```
1677              CALL INTLL(D,IA,IB,IC,ID,D4,GLAT,GLON,CLAT,XLON,IH)
1678              CALL INTLL(T,IA,IB,IC,ID,T4,GLAT,GLON,CLAT,XLON,IH)
1679       C.....RELATIVE LONGITUDE DISPLACEMENT FROM REFERENCE POSITION (IA)
1680              DLON = (XLON - GLON(IA))/(GLON(IB) - GLON(IA))
1681       C.....RELATIVE LATITUDE DISPLACEMENT FROM REFERENCE POSITION(IA)
1682              DLAT = (CLAT - GLAT(IA))/(GLAT(IC) - GLAT(IA))
1683              DPX=P(IB,IH)-P(IA,IH)
1684       C.....DP/DX FOR GEOSTROPHIC WIND EQUATIONS
1685              DPX = DPX + (P(ID,IH) - P(IC,IH) - DPX)*DLAT
1686              DTX = T(IB,IH) - T(IA,IH)
1687       C.....DT/DX FOR THERMAL WIND EQUATIONS
1688              DTX = DTX + (T(ID,IH) - T(IC,IH) - DTX)*DLAT
1689              DPY = P(IC,IH) - P(IA,IH)
1690       C.....DP/DY FOR GEOSTROPHIC WIND EQUATIONS
1691              DPY = DPY + (P(ID,IH) - P(IB,IH) - DPY)*DLON
1692              DTY = T(IC,IH) - T(IA,IH)
1693       C.....DT/DY FOR THERMAL WIND EQUATIONS
1694              DTY = DTY + (T(ID,IH) - T(IB,IH) - DTY)*DLON
1695              IF(NG.GT.9) GO TO 315
1696              DPX=DPX/9.
1697              DTX=DTX/9.
1698              DPY=DPY/3.
1699              DTY=DTY/3.
1700        315   IF(ABS(CLAT).GT.18) GO TO 312
1701              DPY=DPY*5./12
1702              DTY=DTY*5./12
1703        312   IF (NG.GT.9) GO TO 310
1704              DPXX = 0.
1705              DPYY = 0.
1706              DPXY = 0.
1707              RETURN
1708        310   DPXY = P(ID,IH) - P(IC,IH) - P(IB,IH) + P(IA,IH)
1709              IF (MOD(IB,4) .EQ.0) GO TO 320
1710              I1 = IA
1711              I2 = IB + 1
1712              I3 = IC
1713              I4 = ID + 1
1714              SX=1.
1715              GO TO 330
1716        320   I1 = IA - 1
1717              I2 = IB
1718              I3 = IC - 1
1719              I4 = ID
1720              SX=-1.
1721        330   IF(LAX(I1).NE.LAX(IA).OR.LAX(I2).NE.LAX(IA).OR.LAX(I3).NE.
1722             * LAX(IA).OR.LAX(I4).NE.LAX(IA)) GO TO 360
1723              DPXX = P(I2,IH) - P(I1,IH)
1724              DPXX = DPXX + (P(I4,IH) - P(I3,IH) - DPXX)*DLAT
1725              IF (IC.GT.12) GO TO 340
1726              I1 = IA
1727              I2 = IC + 4
1728              I3 = IB
1729              I4 = ID + 4
1730              SY=1.
1731              GO TO 350
1732        340   I1 = IA - 4
1733              I2 = IC
1734              I3 = IB - 4
1735              I4 = ID
1736              SY=-1.
1737        350   IF(LAX(I1).NE.LAX(IA).OR.LAX(I2).NE.LAX(IA).OR.LAX(I3).NE.
1738             * LAX(IA).OR.LAX(I4).NE.LAX(IA)) GO TO 360
1739              DPYY = P(I2,IH) - P(I1,IH)
1740              DPYY = DPYY + (P(I4,IH) - P(I3,IH) - DPYY)*DLON
```

241

```
1741          DPXX =(DPXX - 2.*DPX )*SX
1742          DPYY =(DPYY - 2.*DPY )*SY
1743          RETURN
1744    360   DPXX = 0.
1745          DPYY = 0.
1746          DPXY = 0.
1747          RETURN
1748          END
1749          SUBROUTINE JAC(Z,TZ,DENS)
1750          COMMON/IOTEMP/IOTEM1,IOTEM2,IUG,IUN,DD,XMJD,PHI1,PHI,
1751        .             NSAME,RP1, RD1, RT1, SP1, SD1, ST1, RU1, RV1, SU1, SV1,
1752       $  MN, IDA, IYR, H1, PHI1R,THET1R,G,RI,H,PHIR,THETR,F10,F10B,AP,
1753       $  IHR,MIN,NMORE,DX,HL,VL,DZ,DUMMY(25)
1754          COMMON/COMJAC/XLAT,XLONG,SDA,SHA,DY,Y,T,EM
1755          DIMENSION ALPHA(6),EI(6),DI(6),      B(7),DIT(6)
1756          QQ = 100.
1757          DATA ALPHA/0.0,0.0,0.0,0.0,-0.38,0.0/
1758          DATA EI/28.0134,31.9988,15.9994,39.948,4.0026,1.00797/
1759          DATA B/28.15204,-0.085586,1.284E-04,-1.0056E-05,-1.021E-05,
1760       11.5044E-06,9.9826E-08/
1761          AV=6.02257E23
1762          QN=.78110
1763          QO2=.20955
1764          QA=.009343
1765          QHE = 1.289E-5
1766          FK=8.31432
1767    C
1768    C     TEMPERATURE AT Z = 125 KM, EQ. 9
1769    C
1770          TX=444.3807+.02385*T -392.8292*EXP(-.0021357*T)
1771          A2=2.*(T-TX)/3.14159265
1772    C
1773    C
1774          DIT(6)=0.
1775          M=10
1776          EPS=.0001
1777    C
1778    C     TEMPERATURE FOR 90%Z%125, EQ. 10
1779    C
1780          T1=1.9*(TX-183.)/35.
1781          T4=3.*(TX-183.-2.*T1*35./3.)/(35.**4)
1782          T3=-T1/(3.*35.**2)+4.*T4*35./3.
1783          TZ=TX+T1*(Z-125.)+T3*(Z-125.)**3+T4*(Z-125.)**4
1784          IF (Z-105.) 43,43,40
1785    C
1786    C     MEAN MOLECULAR WEIGHT FOR 90%Z%105, EQ. 1
1787    C
1788     43   Z2 = Z - QQ
1789          EM=B(1)+B(2)*Z2+B(3)*Z2**2+B(4)*Z2**3+B(5)*Z2**4+B(6)*Z2**5
1790       1+B(7)*Z2**6
1791          D=Z
1792     70   CONTINUE
1793    C
1794    C     INTEGRATION OF EQ. 5 FOR DENSITY BETWEEN 90%Z%105
1795    C
1796          A=90.
1797          FA=B(1)+B(2)*(A-QQ)+B(3)*(A-QQ)**2+B(4)*(A-QQ)**3+B(5)*(A-QQ)**4
1798       1+B(6)*(A-QQ)**5 +B(7)*(A-QQ)**6
1799          FA=FA*9.80655/((1.+A/6.356766E+3)**2)
1800          FA=FA/(TX+T1*(A-125.)+T3*(A-125.)**3 +T4*(A-125.)**4)
1801          FD=B(1)+B(2)*(D-QQ)+B(3)*(D-QQ)**2+B(4)*(D-QQ)**3+B(5)*(D-QQ)**4
1802       1+B(6)*(D-QQ)**5 +B(7)*(D-QQ)**6
1803          FD=FD*9.80665/((1.+D/6.356766E+3)**2)
1804          FD=FD/(TX+T1*(D-125.)+T3*(D-125.)**3 +T4*(D-125.)**4)
```

242

```
1805   C      SRQ4, SIMPSONS RULE QUADRATURE  -  G.F.KUNCIR
1806   C      DEFINITIONS  -
1807   C        A - LOWER LIMIT OF INTEGRATION
1808   C        D - UPPER LIMIT OF INTEGRATION
1809   C        FUNC - INTEGRAND FUNCTION SUBPROGRAM
1810   C        EPS - RELATIVE ERROR CONVERGENCE CRITERION
1811   C        M - MAXIMUM NUMBER OF INTEGRATIONS
1812   C        R - RESULT OF INTEGRATION
1813   C        N - NUMBER OF INTEGRATIONS9RIQ&IRID TO FIND R
1814   C
1315          NINT = 1
1816          N=0
1817          PREV=0.
1818          SONE=(D-A)*(FA+FD)/2.
1819   71     N=N+1
1820          IF (N-M) 72,72,75
1821   72     NINT = 2 * NINT
1822          STWO=0.
1823          DEL=(D-A)/FLOAT(NINT)
1824          DO 73 I=1,NINT,2
1825          X=A+DEL*FLOAT(I)
1826          FX=B(1)+B(2)*(X-QQ)+B(3)*(X-QQ)**2+B(4)*(X-QQ)**3+B(5)*(X-QQ)**4
1827         1+B(6)*(X-QQ)**5 +B(7)*(X-QQ)**6
1828          FX=FX*9.80665/((1.+X/6.356766E+3)**2)
1829          FX=FX/(TX+T1*(V-125.)+T3*(X-125.)**3 +T4*(X-125.)**4)
1830   73     STWO=STWO+FX
1831          CUR=SONE+4.*DEL*STWO
1832          IF (EPS*ABS(CUR)-ABS(CUR-PREV)) 74,75,75
1833   74     PREV=CUR
1834          SONE=(SONE+CUR)/4.
1835          GO TO 71
1836   75     R=CUR/3
1837          IF (Z-105.) 44,76,44
1838     44   IF (D-105.) 76,55,76
1839   C
1840   C      DENSITY FOR 90%Z%105
1841   C
1842   76     DENS=3.46E-9*183.*EM*EXP(-R/FK)/(TZ*28.878)
1843          DL=ALOG10(DENS)
1844          PAR=AV*DENS/EM
1845          AN=ALOG10(QN*EM*PAR/28.96)
1846          AA=ALOG10(QA*EM*PAR/28.96)
1847          AHE=ALOG10(QHE*EM*PAR/28.96)
1848          AO=ALOG10(2.*PAR*(1.-EM/28.96))
1849          AO2=ALOG10(PAR*(EM*(1.+QO2)/28.96-1.))
1850          AH=-0.
1851          RETURN
1852   C
1853   C      TEMPERATURE AND MEAN MOLECULAR WEIGHT AT Z=105 KM
1854   C
1855     40   Z3=105.
1856          TZ3=TX+T1*(Z3-125.)+T3*(Z3-125)**3+T4*(Z3-125)**4
1857          ZM3=B(1)+B(2)* 5.+B(3)* 25.+B(4)* 125.+B(5)* 5.**4.+B(6)* 5.**5.
1858         1+B(7)* 5.**6.
1859          D=105.
1860          GO TO 70
1861   C
1862   C      DENSITY  AT Z=105 KM
1863   C
1864   55     DEN1=3.46E-9*183.*ZM3*EXP(-R/FK)/(TZ3*28.878)
1865          PAR=AV*DEN1/ZM3
1866          DI(1)=QN*ZM3*PAR/28.96
1867          DI(2)=PAR*(ZM3*(1.+QO2)/28.96-1.)
1868          DI(3)=2.*PAR*(1.-ZM3/28.96)
```

```
1869              DI(4)=QA*ZM3*PAR/28.96
1870              DI(5)=QHE*ZM3*PAR/28.96
1871              IF(Z-125.) 56,56,90
1872     56       CONTINUE
1873     C
1874     C        INTEGRATION OF EQ. 6 FOR DENSITY ABOVE 105 KM
1875     C
1876              R=0.
1877              D1=125.
1878              A1=105.
1879    400       CONTINUE
1880              FA1=9.80665/((1.+A1/6.356766E+3)**2)
1881              FA1=FA1/(TX+T1*(A1-125.)+T3*(A1-125.)**3+T4*(A1-125.)**4)
1882              FD1=9.80665/((1.+D1/6.356766E+3)**2)
1883              IF(D1-125.) 45,45,50
1884      45 FD1=FD1/(TX+T1*(D1-125.)+T3*(D1-125.)**3+T4*(D1-125.)**4)
1885              GO TO 51
1886     50       FD1=FD1/(TX+A2*ATAN(T1*(D1-125.)*(1.+4.5E-6*(D1-125.)**2.5)/A2))
1887              TZ=TX+A2*ATAN(T1*(Z-125.)*(1.+4.5E-6*(Z-125.)**2.5)/A2)
1888     51       N=0
1889              NINT = 1
1890              PREV=0
1891              SONE=(D1-A1)*(FA1+FD1)/2.
1892     81       N=N+1
1893              IF (N-M) 82,82,85
1894     82       NINT = 2 * NINT
1895              STWO=0.
1896              DEL=(D1-A1)/FLOAT(NINT)
1897              DO 83 I=1,NINT,2
1898              X1=A1+DEL*FLOAT(I)
1899              FX1=9.80665/((1.+X1/6.356766E+3)**2)
1900              IF(X1-125.) 46,46,52
1901      46 FX1=FX1/(TY-T1*(X1-125.)+T3*(X1-125.)**3+T4*(X1-125.)**4)
1902              GO TO 83
1903     52       FX1=FX1/(TX+A2*ATAN(T1*(X1-125.)*(1.+4.5E-6*(X1-125.)**2.5)/A2))
1904     83       STWO=STWO+FX1
1905              CUR=SONE+4.*DEL*STWO
1906              IF (EPS*ABS(CUR)-ABS(CUR-PREV)) 84,85,85
1907     84       PREV=CUR
1908              SONE=(SONE+CUR)/4.
1909              GO TO 81
1910     85       R=CUR/3.+R
1911              IF(A1.EQ.125.) GO TO 430
1912              D1=Z
1913              A1=125.
1914              GO TO 400
1915    430       CONTINUE
1916     C
1917     C        DENSITY ABOVE 105 KM
1918     C
1919              DO 41  I=1,5
1920              DIT(I)=DI(I)*(TZ3/TZ)**(1.+ALPHA(I))*EXP(-EI(I)*R/FK)
1921      41 CONTINUE
1922              DENS=0
1923              DO 42 I=1,6
1924              DENS=DENS+EI(I)*DIT(I)/AV
1925     42       CONTINUE
1926     C
1927     C        MEAN MOLECULAR WEIGHT FOR Z 105 KM
1928     C
1929              EM=DENS*AV/(DIT(1)+DIT(2)+DIT(3)+DIT(4)+DIT(5)+DIT(6))
1930     C
1931     C        LOG DENSITY
1932     C
```

```
1933          DL=ALOG10(DENS)
1934          AN =ALOG10(DIT(1))
1935          AO2=ALOG10(DIT(2))
1936          AO =ALOG10(DIT(3))
1937          AA =ALOG10(DIT(4))
1938          AHE=ALOG10(DIT(5))
1939          IF(Z-500.) 47,48,48
1940       47 DIT(6)=10.**(-6)
1941       48 AH=ALOG10(DIT(6))
1942          AN =AMAX1(-0., AN)
1943          AO2=AMAX1(-0.,AO2)
1944          AO =AMAX1(-0., AO)
1945          AA =AMAX1(-0., AA)
1946          AHE=AMAX1(-0.,AHE)
1947          AH =AMAX1(-0., AH)
1948          RETURN
1949    C
1950    C     TEMPERATURE AND DENSITY AT Z=500 KM
1951    C
1952       90 S=TX+A2*ATAN(T1*375.*(1.+4.5E-6*375.**2.5)/A2)
1953          DI(6)=10.**(73.13-39.4*ALOG10(S)+5.5*ALOG10(S)*ALOG10(S))
1954          A1=500.
1955          IF(Z-500.) 49,60,60
1956    C
1957    C     INTEGRATION OF EQ. 6 FOR DENSITY FOR Z 125 KM
1958    C
1959       49 A1=Z
1960       60 FA1=9.80665/((1.+A1/6.356766E+3)**2)
1961          FA1=FA1/(TX+A2*ATAN(T1*(A1-125.)*(1.+4.5E-6*(A1-125.)**2.5)/A2))
1962          D1=Z
1963          IF(Z-500.) 61,62,62
1964       61 D1=500.
1965       62 FD1=9.80665/((1.+D1/6.356766E+3)**2)
1966          FD1=FD1/(TX+A2*ATAN(T1*(D1-125.)*(1.+4.5E-6*(D1-125.)**2.5)/A2))
1967          N=0
1968          NINT = 1
1969          PREV=0
1970          SONE=(D1-A1)*(FA1+FD1)/2.
1971       91 N=N+1
1972          IF (N-M) 92,92,95
1973       92 NINT = 2 * NINT
1974          STWO=0.
1975          DEL=(D1-A1)/FLOAT(NINT)
1976          DO 93 I=1,NINT,2
1977          X1=A1+DEL*FLOAT(I)
1978          FX1=9.80665/((1.+X1/6.356766E+3)**2)
1979          FX1=FX1/(TX+A2*ATAN(T1*(X1-125.)*(1.+4.5E-6*(X1-125.)**2.5)/A2))
1980       93 STWO=STWO+FX1
1981          CUR=SONE+4.*DEL*STWO
1982          IF (EPS*ABS(CUR)-ABS(CUR-PREV)) 94,95,95
1983       94 PREV=CUR
1984          SONE=(SONE+CUR)/4.
1985          GO TO 91
1986       95 R=CUR/3.
1987    C
1988    C     TEMPERATURE AT Z 500 KM
1989    C
1990          TZ=TX+A2*ATAN(T1*(Z-125.)*(1.+4.5E-6*(Z-125.)**2.5)/A2)
1991          IF(Z-500.) 63,64,64
1992       63 R=-R
1993    C
1994    C     DENSITY OF HYDROGEN FOR Z 500 KM
1995    C
1996       64 DIT(6)=DI(6)*(S/TZ)*EXP(-EI(6)*R/FK)
```

```
1997            GO TO 56
1998            END
1999            SUBROUTINE JACCH(Z,PHIR,THET,PH,DH,TH)
2000      C
2001      C.... DECLARE APPROPRIATE ARGUMENTS TO BE IN GOULD EXTENDED.  IF A
2002      C     NON-EXTENDED PARAMETER IS DECLARED EXTENDED, THE INTERFACE WILL
2003      C     WORK, ALBEIT WITH A SLIGHT EXECUTION OVERHEAD PENALTY.  LJS.
2004      C
2005            COMMON/COMJAC/XLAT,XLONG,SDA,SHA,DY,R,T,EM
2006            COMMON/IOTEMP/IOTEM1,IOTEM2,IUG,IUN,DD,XMJD,PHI1,PHI,
2007           .           NSAME,RP1, RD1, RT1, SP1, SD1, ST1, RU1, RV1, SU1, SV1,
2008           $ M , IDA, IYR, H1, PHI1R,THET1R,G,RI,H,CLAT,CLON ,F10,F10B,AP,
2009           $ IHR,MIN,NMORE,DX,HL,VL,DZ,DUMMY(25)
2010      C
2011      C     JACCH CALCULATES THE PRESSURE, DENSITY, AND TEMPERATURE AT A
2012      C     POINT IN SPACE ABOVE 90 KM FOR A PARTICULAR TIME
2013      C
2014      C     INPUT
2015      C     Z = HEIGHT IN KM
2016      C     PHIR = LATITUDE IN RADIANS
2017      C     THET = LONGITUDE IN DEGREES   (0 TO 360 DEGREES TURNING WESTWARD)
2018      C     F10 = SOLAR RADIO NOISE FLUX (XE - 22 WATTS/M**2)
2019      C     F10B = 81-DAY AVERAGE F10
2020      C     AP = GEOMAGNETIC INDEX
2021      C     M  = MONTH (FOR YEARLY MEAN VARIABLES M IS SET TO 13)
2022      C     IDA = DAY OF MONTH
2023      C     IYR = YEAR
2024      C     IHR = HOUR OF DAY   (UNIVERSAL TIME)
2025      C     MIN = MINUTE   (UNIVERSAL TIME)
2026      C     XMJD = MEAN JULIAN DAY  (SET EQUAL TO ZERO FOR ANNUAL MEAN)
2027      C     DD = DAY NUMBER WITH RESPECT TO JAN 0 OF YEAR IYR
2028      C
2029      C     OUTPUT
2030      C     PH = PRESSURE IN UNITS OF NT/M**2
2031      C     DH = DENSITY IN UNITS OF KG/M**3
2032      C     TH = TEMPERATURE IN KELVIN DEGREES
2033      C
2034      C     DD = DAY NUMBER WITH RESPECT TO JAN 1 OF YEAR IYR
2035      C
2036      C     REPLACEMENT OF SUBROUTINE VARIABLES TO INSURE NO CHANGES IN THEM
2037      C
2038            R = 0.31
2039            XLAT = PHIR
2040            XLONG = THET
2041            IF (M.EQ.13) GO TO 50
2042      C
2043      C     CALCULATE SOLAR DEC. AND HOUR ANGLE
2044      C
2045            CALL TME
2046      C
2047      C     EXOSPHERIC TEMPERATURE
2048      C
2049            CALL TINF
2050            GO TO 75
2051         50 T = 1000.0
2052      C
2053      C     TEMPERATURE, MOLECULAR WEIGHT, AND DENSITY WITHOUT SEASONAL
2054      C        VARIATIONS
2055      C
2056         75 CALL JAC(Z,TH,DH)
2057            IF (M.EQ.13) GO TO 300
2058            YDA = 365.0
2059            J1 = MOD(IYR,4)
2060            IF (J1.EQ.0) YDA = 366.0
```

```
2061          C1 = SIN((360. / YDA) * 0.0174532925 * (DD + 100.0))
2062          IF (PHIR) 80,70,80
2063      70 C2 = 0.0
2064          GO TO 90
2065      80 C2 = (SIN(PHIR) ** 2) * (PHIR / ABS(PHIR))
2066   C
2067   C      DENSITY WITH SEASONAL VARIATIONS
2068   C
2069      90 Z90 = Z - 90.0
2070          DLRHO = 0.02 * Z90 * EXP(-0.045 * Z90) * C1 * C2
2071          DH = DH * EXP(DLRHO)
2072   C
2073   C      MOLECULAR WEIGHT WITH SEASONAL VARIATION
2074   C
2075          IF (Z - 120.0) 100,100,150
2076     100 EM = EM + 0.006 * Z90 * C1
2077          GO TO 250
2078     150 IF (Z - 230.0) 200,250,250
2079     200 DEM = EXP(-0.02424 * Z90) * (0.0316 * Z90 - 0.0002257 * Z90 * Z90)
2080          EM = EM + DEM * C1*0.5
2081   C
2082   C      TEMPERATURE WITH SEASONAL VARIATIONS
2083   C
2084     250 IF (Z-260.0) 270,300,300
2085     270 Z110 = Z - 110.0
2086          DTH = -2.291753 * Z110 + 0.02154336 * Z110*Z110- 4.1766671E-05 *
2087        $ (Z110 ** 3)
2088          DTH = EXP(-0.290655 * SQRT(ABS(Z110)))* DTH
2089          TH = TH +(DTH * C1 * C2 *TH) / 100.0
2090   C
2091   C      DENSITY IN METRIC UNITS AND PRESSURE CALCULATED
2092   C
2093     300 DH = DH * 1000.0
2094          PH =((DH * 8.31432 * TH) / EM) * 1000.0
2095          RETURN
2096          END
2097          SUBROUTINE NORMAL(D1,D2)
2098   C.....PRODUCES 2 RANDOM NUMBERS, D1, D2, PICKED FROM A NORMAL DIST.
2099   C      WITH ZERO MEAN AND UNIT VARIANCE
2100          REAL L
2101          LOGICAL          OP     ,RST   ,HLD    ,RT     ,ATRM  ,ICEN  ,MDAT
2102          COMMON /RTCDAT/ OP     ,RST   ,HLD    ,RT     ,ATRM  ,ICEN  ,MDAT
2103   C
2104   C.... MODIFIED TO OUTPUT ZERO WHEN RESET FLAG IS TRUE.  THIS AVOIDS
2105   C      OUTPUT BIASES INTRODUCED BY UNCHARACTERISTIC LARGE STEPS
2106   C      INTRODUCED AT PROGRAM INITIALIZATION TIME OR WHEN IC'S CHANGE.
2107   C
2108   C.... MODIFIED   1/10/90   L SCHILLING   NASA/ADFRF.
2109   C
2110      50 CONTINUE
2111   C
2112          X = RAND(0)
2113          Y = 2*RAND(0) - 1
2114   C
2115          XX = X**2
2116          YY = Y**2
2117          S = XX + YY
2118          IF (S.GT.1.0) GO TO 50
2119   C
2120      51 CONTINUE
2121   C
2122          L = SQRT(-2.0*ALOG(RAND(0)))/S
2123   C
2124          D1 = (XX-YY)*L
```

```
2125          D2 = 2.0*X*Y*L
2126   C
2127          IF(RST) THEN
2128              D1=0.0
2129          D2=0.0
2130          ENDIF
2131   C
2132          RETURN
2133          END
2134          SUBROUTINE PDTUV (PSP, DSP, TSP, CLAT, CLON, IH, PS, DS, TS,
2135         $ DPX, DPY, DTX, DTY,DP2X,DP2Y,DPXY)
2136   C
2137   C.... DECLARE APPROPRIATE ARGUMENTS TO BE IN GOULD EXTENDED.  IF A
2138   C     NON-EXTENDED PARAMETER IS DECLARED EXTENDED, THE INTERFACE WILL
2139   C     WORK, ALBEIT WITH A SLIGHT EXECUTION OVERHEAD PENALTY.  LJS.
2140   C
2141   C.....INTERPOLATES STATIONARY PERTURBATIONS ON LATITUDE AND LONGITUDE
2142   C        AT HEIGHT IH
2143          DIMENSION PSP(8,10,12),DSP(8,10,12),TSP(8,10,12)
2144          IF (IH.LT.52) GO TO 10
2145          IF (IH.GT.84) GO TO 20
2146   C      HEIGHT INDEX K
2147          K = ((IH+4)/8) - 4
2148          GO TO 30
2149       10 K = (IH-20)/10
2150          GO TO 30
2151       20 K = 8
2152       30 XLON = CLON
2153          IF (CLON.LT.10.) XLON = 360. + CLON
2154   C      LOWER LONGITUDE INDEX J
2155          J = INT((XLON + 20.)/30.)
2156   C.....DLON - RELATIVE LONGITUDE DEVIATION FROM CORNER REFERENCE LOCATION
2157          DLON = (XLON - 30.*J + 20.)/30.
2158   C      UPPER LONGITUDE INDEX JP
2159          JP = J+1
2160          IF (JP.GT.12) JP=1
2161   C      LOWER LATITUDE INDEX I
2162          I = INT((CLAT + 110.)/20.)
2163   C       UPPER LATITUDE INDEX IP
2164          IP = I+1
2165          IF (IP.GT.10) IP=10
2166   C.....DLAT - RELATIVE LATITUDE DEVIATION FROM CORNER REFERENCE LOCATION
2167          DLAT = (CLAT-20.*I + 110.)/20.
2168   C      PRESSURE LAT-LON INTERPOLATION
2169          PS=PSP(K,I,J)+(PSP(K,IP,J)-PSP(K,I,J))*DLAT+(PSP(K,I,JP)-PSP(K,I,J
2170         1))*DLON+(PSP(K,IP,JP)-PSP(K,I,JP)-PSP(K,IP,J)+PSP(K,I,J))*DLAT*
2171         2DLON
2172   C      DENSITY LAT-LON INTERPOLATION
2173          DS=DSP(K,I,J)+(DSP(K,IP,J)-DSP(K,I,J))*DLAT+(DSP(K,I,JP)-DSP(K,I,J
2174         1))*DLON+(DSP(K,IP,JP)-DSP(K,I,JP)-DSP(K,IP,J)+DSP(K,I,J))*DLAT*
2175         2DLON
2176   C      TEMPERATURE LAT-LON INTERPOLATION
2177          TS=TSP(K,I,J)+(TSP(K,IP,J)-TSP(K,I,J))*DLAT+(TSP(K,I,JP)-TSP(K,I,J
2178         1))*DLON+(TSP(K,IP,JP)-TSP(K,I,JP)-TSP(K,IP,J)+TSP(K,I,J))*DLAT*
2179         2DLON
2180   C.....DPX - DP/DX FOR GEOSTROPHIC WINDS
2181          DPX = (PSP(K,I,J) - PSP(K,I,JP)) / 6.
2182          DPX = DPX + ((PSP(K,IP,J) - PSP(K,IP,JP))/6. - DPX)*DLAT
2183   C.....DPY - DP/DY FOR GEOSTROPHIC WINDS
2184          DPY=(PSP(K,IP,J)-PSP(K,I,J))/4.
2185          DPY = DPY + ((PSP(K,IP,JP) - PSP(K,I,JP))/4. - DPY)*DLON
2186   C.....DTX - DT/DX FOR THERMAL WINDS
2187          DTX = (TSP(K,I,J) - TSP(K,I,JP)) / 6.
2188          DTX = DTX + ((TSP(K,IP,J) - TSP(K,IP,JP))/6. - DTX)*DLAT
```

```
2189   C.....DTY - DT/DY FOR THERMAL WINDS
2190         DTY = (TSP(K,IP,J) - TSP(K,I,J)) / 4.
2191         DTY = DTY + ((TSP(K,IP,JP) - TSP(K,I,JP))/4. - DTY)*DLON
2192         IF (IP.GT.9) GO TO 90
2193         DPXY = (PSP(K,IP,J) - PSP(K,IP,JP) - PSP(K,I,J) + PSP(K,I,JP))/24.
2194         JX = J - 1
2195         IF (JX.LT.1) JX = JX + 12
2196         IY = I - 1
2197         DP2X = (PSP(K,I,JX) - PSP(K,I,JP))/6.
2198         DP2X = DP2X + ((PSP(K,IP,JX) - PSP(K,IP,JP))/6. - DP2X)*DLAT
2199         DP2Y = (PSP(K,IP,J) - PSP(K,IY,J))/4.
2200         DP2Y = DP2Y + ((PSP(K,IP,JP) - PSP(K,IY,JP))/4.- DP2Y)*DLON
2201         RETURN
2202   90    DP2X = 0.
2203         DP2Y = 0.
2204         DPXY = 0.
2205         RETURN
2206         END
2207         SUBROUTINE PERTRB
2208   C>>
2209         REAL*8 XCNT,RNDLX,RNDSX,RNTLX,RNTSX,RNULX,RNUSX,RNVLX,RNVSX
2210         COMMON /PLTOUT/ HWLS  ,HWLL  ,VDS   ,VTS   ,VUS   ,VDL   ,VTL   ,
2211        .                VUL   ,RDS   ,RTS   ,RVS   ,RDL   ,RTL   ,RVL   ,
2212        .                RNDL  ,RNDS  ,RNTL  ,RNTS  ,RNUL  ,RNUS  ,RNVL  ,
2213        .                RNVS  ,RNDLM ,RNDSM ,RNTLM ,RNTSM ,RNULM ,RNUSM ,
2214        .                RNVLM ,RNVSM
2215   C>>
2216         COMMON/IOTEMP/IOTEM1,IOTEM2,IUG,  IUN ,DD,XMJD,PHI1,PHI,NSAME,
2217        $PL1,DL1,TL1,SPL1,SDL1,STL1,UL1,VL1,SUL1,SVL1,MN,IDA,IYR,
2218        1PH,PLAT,
2219        * PLON,G,R,CH,CLAT,CLON,F10,F10B,AP,IHR,MIN,NMORE,DX,HL,VL,DZ,
2220        2B,EPS,IOPP,LOOK,ICT,FLAT,PS1,DS1,TS1,US1,VS1,SPS1,SDS1,
2221        3STS1,SUS1,SVS1,UDS1,VDS1,UDL1,VDL1,UDS2,VDS2,UDL2,VDL2,DUMMY3(1)
2222         COMMON /COMPER/SP2,SD2,ST2,P2,D2,T2,U2,V2,SU2,SV2,CP,
2223        1PS2,DS2,TS2,US2,VS2,
2224        2PL2,DL2,TL2,UL2,VL2,
2225        3SPS2,SDS2,STS2,SUS2,SVS2,
2226        4SPL2,SDL2,STL2,SUL2,SVL2
2227         COMMON/WINCOM/ DUM(11),T,DUMMY2(9)
2228   C>>
2229         DATA XCNT,RNDLX,RNDSX,RNTLX,RNTSX,RNULX,RNUSX,RNVLX,RNVSX /9*0.0/
2230   C>>
2231         DLON = ABS(CLON-PLON)
2232         PI = 3.1415927
2233         IF(DLON.GT.PI) DLON = 2.*PI - DLON
2234         DX  =     R*SQRT((CLAT-PLAT)**2 + (COS(CLAT)*(DLON      ))**2)
2235   C.....DX IS HORIZONTAL DISTANCE BETWEEN POSITIONS PLAT,PLON AND CLAT,CLO
2236         AH = 900.
2237         BH = 6.
2238   C     HORIZONTAL WAVELENGTH, KM
2239         HLL= AH + BH*CH
2240   C>>
2241         HWLL=HLL
2242   C>>
2243         DPHI = (90. - ABS(CLAT)/0.017453293)**2
2244         DHGT = 0.22 + 0.00258*(SQRT(ABS(CH)**3))
2245         IF (DHGT.GT.5.) DHGT = 5.
2246         VDS = (11.0 - 2.102E-4*DPHI)*DHGT
2247         VTS = (3.0 + 5.146E-4*DPHI)*DHGT
2248         VUS = (6.2 - 3.515E-4*DPHI)*DHGT
2249         VDL = (20.7 - 1.346E-3*DPHI)*DHGT
2250         VTL = 7.3*DHGT
2251         VUL = (31.2 - 3.503E-3*DPHI)*DHGT
2252         HLS = 20. + .0125*CH*CH
```

249

```
2253              IF(HLS.GT.400.) HLS = 400.
2254    C>>
2255              HWLS=HLS
2256    C>>
2257              HLS = (DX/HLS)**2
2258              HLL = (DX/HLL)**2
2259              RDS=SQRT(HLS+(DZ/VDS)**2)
2260              IF(RDS.LE.100.)GO TO 10
2261              RDS=0.
2262              GO TO 20
2263    10        RDS=CORREL(RDS)
2264    20        RTS=SQRT(HLS+(DZ/VTS)**2)
2265              IF(RTS.LE.100.)GO TO 30
2266              RTS=0.
2267              GO TO 40
2268    30        RTS=CORREL(RTS)
2269    40        RVS=SQRT(HLS+(DZ/VUS)**2)
2270              IF(RVS.LE.100.)GO TO 50
2271              RVS=0.
2272              GO TO 60
2273    50        RVS=CORREL(RVS)
2274    60        RDL=SQRT(HLL+(DZ/VDL)**2)
2275              IF(RDL.LE.100.)GO TO 70
2276              RDL=0.
2277              GO TO 80
2278    70        RDL=CORREL(RDL)
2279    80        RTL=SQRT(HLL+(DZ/VTL)**2)
2280              IF(RTL.LE.100.)GO TO 90
2281              RTL=0.
2282              GO TO 100
2283    90        RTL=CORREL(RTL)
2284    100       RVL=SQRT(HLL+(DZ/VUL)**2)
2285              IF(RVL.LE.100.)GO TO 110
2286              RVL=0.
2287              GO TO 120
2288    110       RVL=CORREL(RVL)
2289    120       CONTINUE
2290              CALL CORLAT(AS,BS,CS,DS,ES,FS,GS,HS,AIS,AJS,AKS,SPS1,SPS2,SDS1,
2291             1 SDS2,STS1,STS2,SUS1,SUS2,SVS1,SVS2,UDS1,UDS2,VDS1,VDS2,RDS,RTS,
2292             2RVS)
2293              CALL CORLAT(AL,BL,CL,DL,EL,FL,GL,HL,AIL,AJL,AKL,SPL1,SPL2,SDL1,
2294             1 SDL2,STL1,STL2,SUL1,SUL2,SVL1,SVL2,UDL1,UDL2,VDL1,VDL2,
2295             2RDL,RTL,RVL)
2296              CALL NORMAL(ZD,ZT)
2297    C>>
2298              XCNT=XCNT+1.0
2299              RNDS=ZD
2300              RNTS=ZT
2301              RNDSX=RNDSX+RNDS
2302              RNTSX=RNTSX+RNTS
2303              RNDSM=RNDSX/XCNT
2304              RNTSM=RNTSX/XCNT
2305    C>>
2306              DS2=AS*DS1+BS*ZD
2307              TS2=CS*TS1+DS*DS2+ES*ZT
2308              FS2=DS2+TS2
2309              CALL NORMAL(ZD,ZT)
2310    C>>
2311              PNUS=ZD
2312              PNVS=ZT
2313              PNUSX=PNUSX+PNUS
2314              PNVSX=PNVSX+PNVS
2315              PNUSM=PNUSX/XCNT
2316              PNVSM=PNVSX/XCNT
```

```
2317    C>>
2318          US2=FS*US1+GS*DS2+HS*ZD
2319          VS2=AIS*VS1+AJS*DS2+AKS*ZT
2320          CALL NORMAL(ZD,ZT)
2321    C>>
2322          RNDL=ZD
2323          RNTL=ZT
2324          RNDLX=RNDLX+RNDL
2325          RNTLX=RNTLX+RNTL
2326          RNDLM=RNDLX/XCNT
2327          RNTLM=RNTLX/XCNT
2328    C>>
2329          DL2=AL*DL1+BL*ZD
2330          TL2=CL*TL1+DL*DL2+EL*ZT
2331          PL2=DL2+TL2
2332          CALL NORMAL(ZD,ZT)
2333    C>>
2334          RNUL=ZD
2335          RNVL=ZT
2336          PNULX=RNULX+RNUL
2337          RNVLX=RNVLX+RNVL
2338          RNULM=RNULX/XCNT
2339          RNVLM=RNVLX/XCNT
2340    C>>
2341          UL2=FL*UL1+GL*DL2+HL*ZD
2342          VL2=AIL*VL1+AJL*DL2+AKL*ZT
2343          P2=PS2+PL2
2344          D2=DS2+DL2
2345          T2=TS2+TL2
2346          IF(P2.LT.-0.9)P2 = -0.9
2347          IF(D2.LT.-0.9)D2 = -0.9
2348          IF(T2.LT.-0.9)T2 = -0.9
2349          U2=US2+UL2
2350          V2=VS2+VL2
2351          UDL1=UDL2
2352          UDS1=UDS2
2353          VDL1=VDL2
2354          VDS1=VDS2
2355          RETURN
2356          END
2357          SUBROUTINE PHASE(D1,X1,D2,X2,D,X)
2358    C
2359    C.... DECLARE APPROPRIATE ARGUMENTS TO BE IN GOULD EXTENDED.  IF A
2360    C     NON-EXTENDED PARAMETER IS DECLARED EXTENDED, THE INTERFACE WILL
2361    C     WORK, ALBEIT WITH A SLIGHT EXECUTION OVERHEAD PENALTY.  LJS.
2362    C
2363          PER = 870.
2364          IF (X2-X1) 20,10,20
2365    10    D = D1
2366          RETURN
2367    20    DA = D1
2368          DB = D2
2369          PER2 = PER/2.
2370          IF(ABS(DB-DA).LE.PER2)GO TO 30
2371          IF (DA.LT.PER2) DA = DA + PER
2372          IF (DB.LT.PER2) DB = DB + PER
2373    30    DA = DA + (DB - DA)*(X - X1)/(X2 - X1)
2374          IF (DA.GT.PER) DA = DA - PER
2375          IF(DA.LT.0.)DA=DA+PER
2376          D = DA
2377          RETURN
2378          END
2379          SUBROUTINE QBOGEN
2380    C.....COMPUTES QBC VALUES PQ,DQ,TQ,UQ,VQ AT HEIGHT H, LATITUDE PHI
```

251

```
2381    C           ON JULIAN DAY XMJD FROM ARRAYS OF AMPLITUDES PAQ,DAQ,TAQ,
2382    C           UAQ,VAQ AND PHASES PDQ,DDQ,TDQ,UDQ,VDQ.
2383           COMMON/IOTEMP/IOTEM1,IOTEM2,IUG,IUN,DDD,XMJD,PHI1,PHI,
2384       .           NSAME,RP1, RD1, RT1, SP1, SD1, ST1, RU1, RV1, SU1, SV1,
2385       $ MN,  IDA, IYR, H1, PHI1R,THET1R,G,RI,H,PHIR,THETR,F10,F10B,AP,
2386       $ IHR,MIN,NMORE,DX,HL,VL,DZ,DUMMY2(25)
2387           COMMON /PDTCOM/
2388       .                IU4,MONTH,IOPR,PG(18,19),TG(18,19),DG(18,19)
2389       . ,PSP(8,10,12)
2390       . ,DSP(8,10,12),TSP(8,10,12),PAQ(17,5),DAQ(17,5),TAQ(17,5),
2391       . PDQ(17,5),DDQ(17,5),TDQ(17,5),PR(20,10),DR(20,10),TR(20,10),
2392       .UAQ(17,5),VAQ(17,5),UDQ(17,5),VDQ(17,5),UR(25,10),VR(25,10)
2393       . ,PQ,DQ,TQ,UQ,VQ
2394       $ ,PA,DA,TA,UA,VA,IOPQ,DUMMY(2250)
2395           IF (XMJD.GT.0.AND.IOPQ.EQ.1) GO TO 10
2396    C      SETS QBO VALUES TO ZERO FOR ANNUAL MEAN
2397           PQ =0.
2398           DQ=0.
2399           TQ=0.
2400           UQ=0.
2401           VQ=0.
2402           RETURN
2403    C      LOWER HEIGHT INDEX
2404        10 IH = INT((H-5.)/5.)
2405           IF (IH.LT.1) IH=1
2406    C      UPPER HEIGHT INDEX
2407           IP = IH + 1
2408           IF (IP.GT.17) IP = 17
2409           PHA = ABS(PHI)
2410    C    LOWER LATITUDE INDEX
2411
2412           JL = INT((  PHA    + 10.)/20.)
2413    C      UPPER LATITUDE INDEX
2414           JP = JL + 1
2415           IF (JL.LE.0) JL=1
2416           IF (JP.GT.5) JP=5
2417    C      JULIAN DAY FOR JAN 0, 1966
2418           XMJDO = 2439126
2419    C      TIME RELATIVE TO JAN 0, 1966
2420           TMJD = XMJD-XMJDO
2421    C      2*PI/PERIOD,PERIOD = 870 DAYS
2422           PER = 870.
2423           TP = 6.2831853/PER
2424    C      LOWER HEIGHT
2425           HI = 5. + 5.*IH
2426    C      LOWER LATITUDE
2427           PHIJ = 20.*JL - 10.
2428    C      UPPER LATITUDE
2429           PHIP = 20.*JP-10.
2430    C.....INTERPOLATES QBO P,D,T AMPLITUDE ON LATITUDE AT LOWER HEIGHT
2431           CALL INTERZ(PAQ(IH,JL),DAQ(IH,JL),TAQ(IH,JL),PHIJ,PAQ(IH,JP),
2432          1DAQ(IH,JP),TAQ(IH,JP),PHIP,PA1,DA1,TA1,PHA)
2433    C      UPPER HEIGHT
2434           HP = 5.+5.*IP
2435    C.....INTERPOLATES QBO P,D,T AMPLITUDE ON LATITUDE AT UPPER HEIGHT
2436           CALL INTERZ(PAQ(IP,JL),DAQ(IP,JL),TAQ(IP,JL),PHIJ,PAQ(IP,JP),
2437          2DAQ(IP,JP),TAQ(IP,JP),PHIP,PA2,DA2,TA2,PHA)
2438    C.....INTERPOLATES QBO P,D,T AMPLITUDE ON HEIGHT AT LATITUDE PHI
2439           CALL INTERZ(PA1,DA1,TA1,HI,PA2,DA2,TA2,HP,PA,DA,TA,H)
2440    C.....INTERPOLATES QBO P,D,T,U,V PHASE ON LATITUDE AND HEIGHT
2441           CALL PHASE(PDQ(IH,JL),PHIJ,PDQ(IH,JP),PHIP,PD1,PHA)
2442           CALL PHASE(DDQ(IH,JL),PHIJ,DDQ(IH,JP),PHIP,DD1,PHA)
2443           CALL PHASE(TDQ(IH,JL),PHIJ,TDQ(IH,JP),PHIP,TD1,PHA)
2444           CALL PHASE(PDQ(IP,JL),PHIJ,PDQ(IP,JP),PHIP,PD2,PHA)
```

252

```
2445          CALL PHASE(DDQ(IP,JL),PHIJ,DDQ(IP,JP),PHIP,DD2,PHA)
2446          CALL PHASE(TDQ(IP,JL),PHIJ,TDQ(IP,JP),PHIP,TD2,PHA)
2447          CALL PHASE(PD1,HI,PD2,HP,PD,H)
2448          CALL PHASE(DD1,HI,DD2,HP,DD,H)
2449          CALL PHASE(TD1,HI,TD2,HP,TD,H)
2450          CALL PHASE(UDQ(IH,JL),PHIJ,UDQ(IH,JP),PHIP,UD1,PHA)
2451          CALL PHASE(VDQ(IH,JL),PHIJ,VDQ(IH,JP),PHIP,VD1,PHA)
2452          CALL PHASE(UDQ(IP,JL),PHIJ,UDQ(IP,JP),PHIP,UD2,PHA)
2453          CALL PHASE(VDQ(IP,JL),PHIJ,VDQ(IP,JP),PHIP,VD2,PHA)
2454          CALL PHASE(UD1,HI,UD2,HP,UD,H)
2455          CALL PHASE(VD1,HI,VD2,HP,VD,H)
2456   C.....INTERPOLATES QBO WIND AMPLITUDE ON LATITUDE AT LOWER HEIGHT
2457          CALL INTERW(UAQ(IH,JL),VAQ(IH,JL),PHIJ,UAQ(IH,JP),VAQ(IH,JP),
2458      5PHIP,UA1,VA1,PHA)
2459   C.....INTERPOLATES QBO WIND AMPLITUDES ON LATITUDE AT UPPER HEIGHT
2460          CALL INTERW(UAQ(IP,JL),VAQ(IP,JL),PHIJ,UAQ(IP,JP),VAQ(IP,JP),
2461      6PHIP,UA2,VA2,PHA)
2462   C.....INTERPOLATES QBO WIND AMPLITUDES ON HEIGHT AT LATITUDE PHI
2463          CALL INTERW(UA1,VA1,HI,UA2,VA2,HP,UA,VA,H)
2464   C.....EVALUATES QBO VALUES FROM INTERPOLATED AMPLITUDES AND PHASES
2465          PQ=PA*COS(TP*(TMJD-PD))
2466          DQ=DA*COS(TP*(TMJD-DD))
2467          TQ=TA*COS(TP*(TMJD-TD))
2468          UQ=UA*COS(TP*(TMJD-UD))
2469          VQ=VA*COS(TP*(TMJD-VD))
2470          RETURN
2471          END
2472          FUNCTION RAND(X0)
2473   C.....PRODUCES A RANDOM NUMBER FROM A UNIFORM DIST. FROM 0 TO +1
2474          INTEGER X0
2475          DOUBLE PRECISION X
2476          IF (X0.NE.0) X = X0/262144.
2477          X = X*509
2478          X = X - INT(X)
2479          RAND = X
2480          RETURN
2481          END
2482          SUBROUTINE RIG
2483          COMMON/IOTEMP/IOTEM1,IOTEM2,IUG,IUN,DD,XMJD,PHI1,PHI,
2484      .          NSAME,RP1, RD1, RT1, SP1, SD1, ST1, RU1, RV1, SU1, SV1,
2485      $ MN, IDA, IYR, H1, PHI1R,THET1R,G,R ,H PHIR,THETR,F10,F10B,AP,
2486      $ IHR,MIN,NMORE,DX,HL,VL,DZ,B,EPS,IOPP,LOOK,IET,GLAT,
2487      1RP1S,RD1S,RT1S,RU1S,RV1S,SP1S,SD1S,ST1S,SU1S,SV1S,
2488      2UDS1,VDS1,UDL1,VDL1,UDS2,VDS2,UDL2,VDL2,DUMMY3(1)
2489   C.....GRAVITY G AT H, LATITUDE PHIR (RADIANS)
2490   C.....RADIUS RI FROM CENTER OF EARTH TO HEIGHT H
2491   C.....B = POLAR EARTH RADIUS, EPS = ECCENTRICITY
2492          CPHI2 = COS(PHIR) ** 2
2493   C      EARTH RADIUS
2494          RI = B / SQRT(1. - EPS * CPHI2)
2495   C      C2PHI = COS(2*PHIR)
2496          C2PHI = 2. * CPHI2 - 1.
2497   C      C4PHI = COS(4*PHIR)
2498          C4PHI = 8. * CPHI2 * (CPHI2 - 1.) + 1.
2499   C.....G AT SURFACE
2500          G = 9.80616 * (1. - 0.0026373 * C2PHI + 0.0000059 * C2PHI * C2PHI)
2501   C.....EFFECTIVE RADIUS
2502          RE = 2. * G / (3.085462E-3 + C2PHI * 2.27E-6 - C4PHI * 2.E-9)
2503   C      G AT HEIGHT H
2504          G = G / (1. + (H  RE)) ** 2
2505   C      RADIUS AT HEIGHT H
2506          RI = RI + H
2507          END
2508          SUBROUTINE RTERP(H,PHI,PR,DR,TR,P,D,T)
```

253

```
2509    C
2510    C.....COMPUTES RANDOM PERTURBATION STANDARD DEVIATIONS P,D,T AT
2511    C         HEIGHT H (KM), LATITUDE PHI(DEGREES) FROM SIGMA ARRAYS
2512    C         PR,DR,AND TR
2513    C
2514    C.... DECLARE APPROPRIATE ARGUMENTS TO BE IN COULD EXTENDED.  IF A
2515    C      NON-EXTENDED PARAMETER IS DECLARED EXTENDED, THE INTERFACE WILL
2516    C      WORK, ALBEIT WITH A SLIGHT EXECUTION OVERHEAD PENALTY.  LJS.
2517    C
2518          DIMENSION PR(20,10),DR(20,10),TR(20,10)
2519    C.....I = LOWER HEIGHT INDEX
2520          IF (H.LT.95.) I = INT((H-20.)/5.)
2521          IF (H.GE.95.) I = 14 + INT((H-80.)/20.)
2522          IF(I.LT.1)I=1
2523          IP = I+1
2524          IF (IP.GT.20) IP = 20
2525    C      LOWER LATITUDE INDEX
2526          J = INT((PHI + 110.)/20.)
2527          JP = J+1
2528          IF (JP.GT.10) JP=10
2529          IF (I.GT.14) GO TO 10
2530    C      LOWER HEIGHT FOR PR,TR,DR  ARRAYS
2531          Z1=5.*I+20.
2532          GO TO 20
2533       10 Z1=20.*(I-10)
2534       20 IF (IP.GT.14) GO TO 30
2535    C      UPPER HEIGHT FOR PR,DR,TR ARRAYS
2536          Z2=5.*IP+20.
2537          GO TO 40
2538       30 Z2=20.*(IP-10)
2539       40 PHI1=-110.+20.*J
2540          PHI2=-110.+20.*JP
2541    C.....INTERPOLATE ON LATITUDE AT LOWER HEIGHT
2542          CALL INTERZ(PR(I,J),DR(I,J),TR(I,J),PHI1,PR(I,JP),DR(I,JP),
2543         1           TR(I,JP),PHI2,P1,D1,T1,PHI)
2544    C.....INTERPOLATE ON LATITUDE AT UPPER HEIGHT
2545          CALL INTERZ(PR(IP,J),DR(IP,J),TR(IP,J),PHI1,PR(IP,JP),DR(IP,JP),
2546         1           TR(IP,JP),PHI2,P2,D2,T2,PHI)
2547    C.....INTERPOLATION ON HEIGHT USING LATITUDE INTERPOLATED VALUES
2548          CALL INTERZ(P1,D1,T1,Z1,P2,D2,T2,Z2,P,D,T,H)
2549          RETURN
2550          END
2551          SUBROUTINE SCIMOD(NPOP)
2552    C.....COMPUTES VALUES P,D,T,U,V AND SHEAR DUH,DVH FROM INPUT AND
2553    C         ARRAYS IN COMMON PDTCOM.  INPUT TO SCIMOD IS:
2554    C         G = GRAVITY AT POSITION       RI = RADIUS AT HEIGHT H
2555    C         PHIR = LATITUDE (RADIANS)     THETR = LONGITUDE (RADIANS)
2556    C         F10 = F10.7 SOLAR FLUX        F10B = MEAN F10.7 FLUX
2557    C         AP = SOLAR-GEOMAGNETIC A SUB P INDEX
2558    C         MN/IDA/IYR = DATA (IYR = FULL YEAR-1900)
2559    C         IHR*MIN = TIME               H1 = PREVIOUS HEIGHT
2560    C         PHI1R = PREVIOUS LATITUDE    THET1R = PREVIOUS LONGITUDE
2561    C         RP1,RD1,RT1 = PREVIOUS RANDOM PERTURBATIONS
2562    C         SP1,SD1,ST1 = PREVIOUS RANDOM STANDARD DEVIATIONS (SIGMAS)
2563    C         RU1,RV1 = PREVIOUS RANDOM WINDS
2564    C         SU1,SV1 = PREVIOUS RANDOM WIND SIGMAS
2565          COMMON/IPRTF/ IFRT
2566          COMMON/IOTEMP/IOTEM1,IOTEM2,IUG, IUN ,DE,XMJD,PHI1,PHI,
2567         .NSAME,RP1L,RD1L,RT1L,SP1L,SD1L,ST1L,PU1L,PV1L,SU1L,SV1L,
2568         $ MN, IDA, IYR, H1, PHI1R,THET1R,G,PI,H,FHIP,THETP,F10,F10B,AP,
2569         .  IHR,MIN,NMORE,DX,HL,VL,DZ,B,EPS,IOFF,LOOK,IET,FLAT,
2570         1RP1S,RD1S,RT1S,PT1S,RV1S,SP1S,SD1S,ST1S,SU1S,SV1S,
2571         2UDS1,VDS1,UDL1,VDL1,UDS2,VDS2,UDL2,VDL2,DUMMY3(1)
2572          COMMON /PDTCOM/
```

```
2573                    IU4,MONTH,IOPR,PG(18,19),TG(18,19),DG(18,19)
2574        . ,PSP(8,10,12)
2575        . ,DSP(8,10,12),TSP(8,10,12),PAQ(17,5),DAQ(17,5),TAQ(17,5),
2576        . PDQ(17,5),DDQ(17,5),TDQ(17,5),PR(20,10),DR(20,10),TR(20,10),
2577        .UAQ(17,5),VAQ(17,5),UDQ(17,5),VDQ(17,5),UR(25,10),VR(25,10), PQ
2578        . ,DQ,TQ,UQ,VQ,PQA,DQA,TQA,UA,VA,IOPQ,
2579        1PLP(25,10),DLP(25,10),TLP(25,10),
2580        2ULP(25,10),VLP(25,10),UDL(25,10),
2581        3VDL(25,10),UDS(25,10),VDS(25,10)
2582         COMMON /C4    /
2583        .          GLAT(16),GLON(16),NG,P4D(16,26),D4D(16,26),T4D(16,26),
2584        . SP4(16,26),SD4(16,26),ST4(16,26),THET1,THET,DUMMY
2585        COMMON/COMPER/SPH,SDH,STH,PRH,DRH,TRH,URH,VRH,SUH,SVH,CP,
2586        1PRHS,DRHS,TRHS,URHS,VRHS,PRHL,DRHL,TRHL,URHL,VRHL,
2587        2SPHS,SDHS,STHS,SUHS,SVHS,SPHL,SDHL,STHL,SUHL,SVHL
2588         COMMON/WINCOM/DH,FCORY,DX5,DY5,DPX,DPY,DPXX,DPXY,DPYY,UGH,VGH,
2589        $   TH,DTX,DTY,DUH,DVH,PH ,UPRE,VPRE,DUPRE,DVPRE
2590        COMMON/CHIC/LA(4,4),NB(2),IWSYM,UCOEF(14,9),VCOEF(14,9)
2591        COMMON /GRAMOT/ PGH    ,DGH    ,TGH    ,UH    ,VH    ,PS    ,DS   ,
2592        .               TS   ,PGHP   ,DGHP   ,TGHP   ,PHP   ,DHP   ,THP   ,
2593        .               PSH   ,DSH   ,TSH   ,WGH
2594        REAL MOLWT
2595        DATA IBLK/1H /,IAST/1H*/
2596 C      FACTOR FOR RADIANS TO DEGREES
2597        FAC = 57.2957795
2598        IWSYM = IBLK
2599        IF(NPOP.NE.0) GO TO 6
2600        UPRE=0.
2601        VPRE=0.
2602        DUPRE=0.
2603        DVPRE=0.
2604 6      PQ=0.
2605        DQ=0.
2606        TQ=0.
2607        PRH=0.
2608        DRH=0.
2609        TRH=0.
2610        URH=0.
2611        VRH=0.
2612        UQ=0.
2613        VQ=0.
2614        PQA=0.
2615        DQA=0.
2616        TQA=0.
2617        UA=0.
2618        VA=0.
2619        PSH=0.
2620        DSH=0.
2621        TSH=0.
2622        MONTH=MN
2623 C      PRESENT LATITUDE, DEG
2624        PHI = PHIR*FAC
2625 C      PRESENT LONGITUDE, DEG
2626        THET = THETR*FAC
2627 C      PREVIOUS LATITUDE, DEG
2628        PHI1 = PHI1P*FAC
2629 C      PREVIOUS LONGITUDE, DEG
2630        THET1 = THET1P*FAC
2631 C.....FCORY = NORTH COMPONENT CORIOLIS FACTOR TIMES DISTANCE FOR
2632 C        5 DEGREES OF LATITUDE
2633        DY5 = 5000.*RI/FAC
2634        DX5 = DY5*COS(PHIR)
2635        FCORY = DY5*SIN(PHIR)/(120.*FAC)
2636 C.....IN JACCHIA OR MIXED GROVES-JACCHIA HEIGHT RANGE
```

```
2637          8 IF(H.GT.90.0) GO TO 10
2638     C.....IN 4-D DATA HEIGHT RANGE
2639            IF (H.LE.25.0) GO TO 500
2640     C       IN GROVES OR MIXED GROVES 4D HEIGHT RANGE
2641            GO TO 200
2642     C.....IN MIXED JACCHIA-GROVES RANGE, NEED TO FAIR DATA
2643     10     IF (H.LT.115.) GO TO 20
2644     C.....FOLLOWING IS THE PURE JACCHIA HEIGHT RANGE SECTION
2645     C.....JACCHIA VALUES AT CURRENT POSITION
2646            CALL JACCH(H,PHIR,THET,PH,DH,TH)
2647            PHIN = PHIR + 5. / FAC
2648            THETE = THET - 5.
2649     C.....JACCHIA VALUES AT CURRENT POSITION+5 DEGREES LAT, FOR DP/DY AND
2650     C        DT/DY
2651            CALL JACCH(H,PHIN,THET,PHN,DHN,THN)
2652     C.....JACCHIA VALUES AT CURRENT POSITION-5 DEGREES LON, FOR DP/DX AND
2653     C        DT/DX
2654            CALL JACCH(H,PHIR,THETE,PHE,DHE,THE)
2655     C      DP/DY FOR GEOSTROPHIC WIND
2656            DPY=PHN-PH
2657     C      DP/DX FOR GEOSTROPHIC WIND
2658            DPX=PHE-PH
2659     C      DT/DX FOR THERMAL WIND SHEAR
2660            DTX = THE - TH
2661     C      DT/DY FOR THERMAL WIND SHEAR
2662            DTY = THN - TH
2663     C       CHANGE NOTATION FOR OUTPUT
2664            PGH=PH
2665            DGH=DH
2666            TGH=TH
2667            CALL WIND
2668            UH = UGH
2669            VH = VGH
2670            HB = H + 5.
2671            CP = 7.*PH/(2.*DH*TH)
2672            CALL JACCH(HB,PHIR,THET,PB,DB,TB)
2673            DTZ = (TB - TH)/5000.
2674     C.....VERTICAL MEAN WIND
2675            WGH = -CP*(UH*DTX/DX5 + VH*DTY/DY5)/(G + CP*DTZ + UH*DUH+VH*DVH)
2676     C       GO TO RANDOM PERTURBATIONS SECTION
2677            GO TO 800
2678     C.....FOLLOWING IS THE MIXED JACCHIA-GROVES HEIGHT RANGE SECTION
2679     C       LOWER HEIGHT INDEX
2680     20     IHA = 5*(INT(H)/5)
2681     C        UPPER HEIGHT INDEX
2682            IHB = IHA + 5
2683     C      LOWER HEIGHT FOR INTERPOLATION
2684            HA = IHA*1.
2685     C      UPPER HEIGHT FOR INTERPOLATION
2686            HB = IHB*1.
2687     C.....JACCHIA VALUES AT LOWER HEIGHT, CURRENT LAT-LON
2688            CALL JACCH(HA,PHIR,THET,PJA,DJA,TJA)
2689            PHIN = PHIR + 5. / FAC
2690            THETE = THET - 5.
2691     C.....JACCHIA VALUES AT LOWER HEIGHT, CURRENT LAT-LON+5 DEGREES
2692     C        LAT. FOR DP/DY AND DT/DY
2693            CALL JACCH(HA,PHIN,THET,PJN,DJN,TJN)
2694     C.....JACCHIA VALUES AT LOWER HEIGHT, CURRENT LAT-LON-5 DEGREES
2695     C        LON, FOR DP/DX. AND DT/DX
2696            CALL JACCH(HA,PHIR,THETE,PJE,DJE,TJE)
2697     C      JACCHIA DP/DY AT LOWER HEIGHT
2698            DPXJA=PJE-PJA
2699     C       JACCHIA DP/DY AT LOWER HEIGHT
2700            DPYJA=PJN-PJA
```

256

```
2701   C       JACCHIA DT/DX AT LOWER HEIGHT
2702           DTXJA = TJE - TJA
2703   C       JACCHIA DT/DY AT LOWER HEIGHT
2704           DTYJA = TJN - TJA
2705   C.....JACCHIA VALUES AT UPPER HEIGHT, CURRENT LAT-LON
2706           CALL JACCH(HB,PHIR,THET,PJB,DJB,TJB)
2707           PHIN = PHIR + 5. / FAC
2708           THETE=THETE-5
2709   C.....JACCHIA VALUES AT UPPER HEIGHT, CURRENT LAT/LON+5 DEGREES
2710   C       LAT, FOR DP/DY AND DT/DY
2711           CALL JACCH(HB,PHIN,THET,PJN,DJN,TJN)
2712   C.....JACCHIA VALUES AT UPPER HEIGHT, CURRENT LAT-LON-5 DEGREES
2713   C       LON, FOR DP/DX AND DT/DX
2714           CALL JACCH(HB,PHIR,THETE,PJE,DJE,TJE)
2715   C       JACCHIA DP/DX FOR GEOSTROPHIC WINDS
2716           DPXJB = PJE - PJB
2717   C       JACCHIA DP/DY FOR GEOSTOPHIC WINDS
2718           DPYJB = PJN - PJB
2719   C       JACCHIA DT/DX FOR THERMAL WIND SHEAR
2720           DTXJB = TJE - TJB
2721   C       JACCHIA DT/DY FOR THERMAL WIND SHEAR
2722           DTYJB = TJN - TJB
2723   C.....GROVES AT LOWER HEIGHT, TO BE FAIRED WITH JACCHIA
2724           CALL GTERP(IHA,PHI,PGA,DGA,TGA,PG,DG,TG,DPYGA,DTYGA,DP2YGA)
2725   C.....GROVES AT UPPER HEIGHT, TO BE FAIRED WITH JACCHIA
2726           CALL GTERP(IHB,PHI,PGB,DGB,TGB,PG,DG,TG,DPYGB,DTYGB,DP2YGB)
2727   C.....FAIRED RESULTS AT LOWER HEIGHT
2728           IHSB = 90
2729           CALL PDTUV(PSP,DSP,TSP,PHI,THET,IHSB,PSH,DSH,TSH,DPXSB,
2730          $ DPYSB,DTXSB,DTYSB,DP2XSB,DP2YSB,DPXYSB)
2731           PGA = PGA*(1. + PSH)
2732           DGA = DGA*(1. + DSH)
2733           TGA = TGA*(1. + TSH)
2734           PGB = PGB*(1. + PSH)
2735           DGB = DGB*(1. + DSH)
2736           TGB = TGB*(1. + TSH)
2737           DTXGA = DTXSB * TGA
2738           DTXGB = DTXSB * TGB
2739           DTYGA = TGA*DTYSB + DTYGA*(1. + TSH + DTYSB)
2740           DTYGB = TGB*DTYSB + DTYGB*(1. + TSH + DTYSB)
2741           DPXGA = DPXSB * PGA
2742           DPXGB = DPXSB * PGB
2743           DPYGA = PGA*DPYSB + DPYGA*(1. + PSH + DPYSB)
2744           DPYGB = PGB*DPYSB + DPYGB*(1. + PSH + DPYSB)
2745           CALL FAIR(PGA,DGA,TGA,PJA,DJA,TJA,IHA,P1,D1,T1,DPXGA,DPYGA,
2746          $ DPXJA,DPYJA,DPXA,DPYA,DTXGA,DTYGA,DTXJA,DTYJA,DTXA,DTYA)
2747   C.....FAIRED RESULTS AT UPPER HEIGHT
2748           CALL FAIR(PGB,DGB,TGB,PJB,DJB,TJB,IHB,P2,D2,T2,DPXGB,DPYGB,
2749          $DPXJB,DPYJB,DPXB,DPYB,DTXGB,DTYGB,DTXJB,DTYJB,DTXB,DTYB)
2750   C.....HEIGHT INTERPOLATION ON FAIRED P,D,T
2751           CALL INTER2(P1,D1,T1,HA,P2,D2,T2,HB,PH,DH,TH,H)
2752   C.....HEIGHT INTERPOLATION ON FAIRED DP/DX,DP/DY
2753           CALL INTERW(DPXA,DPYA,HA,DPXB,DPYB,HB,DPX,DPY,H)
2754   C.....HEIGHT INTERPOLATION ON FAIRED DT/DX,DT/DY
2755           CALL INTERW(DTXA,DTYA,HA,DTXB,DTYB,HB,DTX,DTY,H)
2756   C.....EASTWARD COMPONENT OF GEOSTROPHIC WIND
2757           CALL WIND
2758   C       CHANGE OF VARIABLES FOR OUTPUT
2759           PGH=PH
2760           DGH=DH
2761           TGH=TH
2762           UH = UGH
2763           VH = VGH
2764           CP = 7.*PH/(2.*DH*TH)
```

```
2765            DTZ = (T2 - T1)/5000.
2766    C.....VERTICAL MEAN WIND
2767            WGB = -CP*(UB*DTX/DX5 + VB*DTY/DY5)/(G + CP*DTZ + UB*DUB + VB*DVB)
2768    C       GO TO RANDOM PERTURBATIONS SECTION
2769            GO TO 800
2770    C.....THE FOLLOWING SECTION IS FOR GROVES OR MIXED GROVES 4D HEIGHTS
2771    C       UPPER HEIGHT INDEX
2772      200   IHGB = 5*(INT(H)/5) + 5
2773            IF (IHGB.GT.90) IHGB=90
2774    C       UPPER HEIGHT
2775            HGB = IHGB*1.
2776    C.....GROVES AT UPPER HEIGHT
2777            CALL GTERP(IHGB,PHI,PGB,DGB,TGB,PG,DG,TG,DPYGB,DTYGB,DP2YGB)
2778    C.....UPPER STATIONARY PERTURBATION HEIGHT = 40
2779            IF (H.LT.40.0) GO TO 210
2780    C.....UPPER STATIONARY PERTURBATION HEIGHT = 90
2781            IF (H.GT.84.0) GO TO 220
2782    C.....UPPER STATIONARY PERTURBATION HEIGHT = 52,60,68,76,OR 84
2783            IHSB = 8*((INT(H) + 4)/8) + 4
2784    C.....UPPER STATIONARY PERTURBATION HEIGHT = 52
2785            IF (IHSB.LT.52.0) IHSB = 52
2786            GO TO 230
2787      210   IHSB = 10*(INT(H)/10) + 10
2788            GO TO 230
2789      220   IHSB = 90
2790    C       UPPER STATIONARY PERTURBATION HEIGHT
2791      230   HSB = IHSB*1.
2792    C.....STATIONARY PERTURBATIONS AT UPPER HEIGHT
2793            CALL PDTUV(PSP,DSP,TSP,PHI,THET,IHSB,PSB,DSB,TSB,DPXSB,DPYSB,
2794          $ DTXSB,DTYSB,DP2XSB,DP2YSB,DPXYSB)
2795    C       MIXED GROVES 4D SECTION
2796            IF (H.LT.30.0) GO TO 300
2797    C       LOWER HEIGHT INDEX
2798            IHGA = IHGB - 5
2799    C       LOWER HEIGHT INDEX
2800            HGA = IHGA*1.
2801    C.....GROVES AT LOWER HEIGHT
2802            CALL GTERP(IHGA,PHI,PGA,DGA,TGA,PG,DG,TG,DPYGA,DTYGA,DP2YGA)
2803    C.....LOWER STATIONARY PERTURBATION HEIGHT = 30
2804            IF (H.LT.40.0) GO TO 240
2805    C.....LOWER STATIONARY PERTURBATION HEIGHT = 52,60,68,76, OR 84
2806            IHSA = 8*((INT(H) + 4)/8) - 4
2807    C.....LOWER STATIONARY PERTURBATIONS HEIGHT = 40
2808            IF (IHSA.LT.52.0) IHSA = 40
2809            GO TO 250
2810      240   IHSA = 30
2811    C       LOWER STATIONARY PERTURBATION HEIGHT
2812      250   HSA = IHSA*1.
2813    C.....STATIONARY PERTURBATIONS AT LOWER HEIGHT
2814            CALL PDTUV(PSP,DSP,TSP,PHI,THET,IHSA,PSA,DSA,TSA,DPXSA,DPYSA,
2815          $ DTXSA,DTYSA,DP2XSA,DP2YSA,DPXYSA)
2816    C.....GROVES VALUES HEIGHT INTERPOLATIONS
2817            CALL INTER2(PGA,DGA,TGA,HGA,PGB,DGB,TGB,HGB,PGH,DGH,TGH,H)
2818    C.....STATIONARY PERTURBATION HEIGHT INTERPOLATION
2819            CALL INTER2(PSA,DSA,TSA,HSA,PSB,DSB,TSB,HSB,PSH,DSH,TSH,H)
2820    C       QUASI-BIENNIAL VALUES
2821            CALL QBOGEN
2822    C.....HEIGHT INTERPOLATION OF GROVES DP/DY, DT/DY, AND D2P/DY2
2823            CALL INTER2(DPYGA,DTYGA,DP2YGA,HGA,DPYGB,DTYGB,DP2YGB,HGB,DPYG,
2824          $   DTYG,DP2YG,H)
2825    C.....HEIGHT INTERPOLATION OF STATIONARY PERTURBATION DP/DX AND DP/DY
2826            CALL INTERW(DPXSA,DPYSA,HSA,DPXSB,DPYSB,HSB,DPXS,DPYS,H)
2827    C.....HEIGHT INTERPOLATION OF STATIONARY PERTURBATION DT/DX AND DT/DY
2828            CALL INTERW(DTXSA,DTYSA,HSA,DTXSB,DTYSB,HSB,DTXS,DTYS,H)
```

258

```
2829   C.....HEIGHT INTERPOLATION OF STATIONARY PERTURBATION D2P/DX2,D2P/DY2,
2830   C                AND D2P/DXDY
2831         CALL INTERZ(DP2XSA,DP2YSA,DPXYSA,HSA,DP2XSB,DP2YSB,DPXYSB,HSB,
2832      $         DP2XS,DP2YS,DPXYS,H)
2833   C.....UNPERTURBED (MONTHLY MEAN) VALUES FOR OUTPUT
2834         TGH = TGH * (1. + TSH)
2835         PGH = PGH * (1. + PSH)
2836         DGH = DGH * (1. + DSH)
2837   C     TOTAL DT/DX
2838         DTX =        DTXS * TGH
2839   C     TOTAL DT/DY
2840         DTY = TGH*DTYS + DTYG*(1. + TSH + DTYS)
2841   C     TOTAL DP/DX
2842         DPX =        DPXS * PGH
2843   C     TOTAL DP/DY
2844         DPY = PGH*DPYS + DPYG*(1. + PSH + DPYS)
2845   C     D2P/DX2
2846         DPXX = PGH*(2.*DPXS - DP2XS)
2847         DPYY = PGH*(2.*DPYS - DP2YS) +  (2.*DPYG - DP2YG)*(1. +PSH+DPYS)
2848      $        - (DPYG - DP2YG)*DP2YS
2849   C     D2P/DXDY
2850         DPXY = (PGH + DPYG)*DPXYS + DPYG*DPXS
2851   C.....UNPERTURBED VALUES PLUS QBO PERTURBATIONS
2852         PH = (1. + PQ) * PGH
2853         DH = DGH * (1. + DQ)
2854         TH = (1. + TQ) * TGH
2855         CALL WIND
2856   C     GEOSTROPHIC WIND PLUS QBO WIND PERTURBATIONS
2857         UH=UGH+UQ
2858         VH=VGH+VQ
2859         CP = 7.*PGH/(2.*DGH*TGH)
2860         DTZ = (TGB*(1.+TSB) - TGA*(1.+TSA))/5000.
2861   C.....VERTICAL MEAN WIND
2862         WGH=-CP*(UGH*DTX/DX5+VGH*DTY/DY5)/(G+CP*DTZ+VGH*DUH+VGH*DVH)
2863   C       GO TO RANDOM PERTURBATIONS SECTION
2864         GO TO 800
2865   C.....THE FOLLOWING IS THE MIXED GROVES 4D SECTION
2866   C.....GENERATE GRID OF 4D PROFILES IF PREVIOUS HEIGHT GE 30
2867    300   IF (H1.GE.30..OR.LOOK.EQ.1) CALL GEN4D
2868   C300   IF (H1.GE.30..OR.LOOK.EQ.1) CALL USGRID
2869   C
2870    310   CONTINUE
2871   C.....LAT-LON INTERPOLATION OF 4D DATA AT 25 KM
2872         CALL INTER4(          PHI,THET,25,    P4D,D4D,T4D,P4A,D4A,T4A,
2873      $ DPX4,DPY4,DTX4,DTY4,DPXXA,DPYYA,DPXYA)
2874   C       GROVES PLUS STATIONARY PERTURBATIONS
2875         PB = PGB*(1. + PSB)
2876   C       P,D,T
2877         DB = DGB*(1. + DSB)
2878         TB = TGB*(1. + TSB)
2879         DPXB = PGB*DPXSB
2880         DPYB = PGB*DPYSB + DPYGB*(1. + PSB + DPYSB)
2881         DPXXB = PGB*(2.*DPXSB - DP2XSB)
2882         DPYYB = PGB*(2.*DPYSB - DP2YSB) + (2.*DPYGB - DP2YGB)*
2883      $ (1. + PSB + DPYSB) - (DPYGB - DP2YGB)*DP2YSB
2884         DPXYB = (PGB + DPYGB)*DPXYSB + DPYGB*DPXSB
2885         DTXB = TGB*DTXSB
2886         DTYB = TGB*DTYSB + DTYGB*(1. + TSB + DTYSB)
2887   C.....HEIGHT INTERPOLATION BETWEEN 4D AT 25 AND GROVES AT UPPER HEIGHT
2888   C       DP/DX AND DP/DY
2889         CALL INTERW(DPX4,DPY4,25.,DPXB,DPYB,HSB,DPX,DPY,H)
2890   C.....HEIGHT INTERPOLATION BETWEEN 4D AT 25 AND GROVES AT UPPER HEIGHT
2891   C       P,D,T
2892         CALL INTER2(P4A,D4A,T4A,25.,PB,DB,TB,HGB,PGH,DGH,TGH,H)
```

259

```
2893    C.....HEIGHT INTERPOLATION BETWEEN 4D AT 25 AND GROVES AT UPPER HEIGHT
2894    C        DT/DX AND DT/DY
2895            CALL INTERW(DTX4,DTY4,25.,DTXB,DTYB,HSB,DTX,DTY,H)
2896    C.....HEIGHT INTERPOLATION BETWEEN 4D AT 25 KM AND GROVES AT UPPER
2897    C       HEIGHT D2P/DX2, D2P/DY2, AND D2P/DXDY
2898            CALL INTERZ(DPXXA,DPYYA,DPXYA,25.,DPXXB,DPYYB,DPXYB,HGB,DPXX,
2899          $      DPYY,DPXY,H)
2900            IF (IOPQ.EQ.2) GO TO 350
2901    C       QUASI BIENNIAL PERTURBATIONS
2902            CALL QBOGEN
2903    C       ADD QBO PERTURBATIONS TO P,D,T
2904      350 PH=PGH*(1.+PQ)
2905            DH=DGH*(1.+DQ)
2906            TH=TGH*(1.+TQ)
2907            CALL WIND
2908    C       ADD QBO WIND PERTURBATIONS
2909            UH=UGH+UQ
2910            VH=VGH+VQ
2911            CP = 7.*PGH/(2.*DGH*TGH)
2912            DTZ = (TB - T4A)/(1000.*(HGB - 25.))
2913    C.....VERTICAL MEAN WIND
2914            WGH=-CP*(UGH*DTX/DX5+VGH*DTY/DY5)/(G+CP*DTZ+UGH*DUH+VGH*DVH)
2915    C       GO TO RANDOM PERTURBATIONS SECTION
2916    C:: 2000 FORMAT(1H ,'LATITUDE',/16F8.3)
2917    C:: 2001 FORMAT(1H ,'LONGITUDE',/16F8.3,/' PRESSURE')
2918    C:: 2002 FORMAT(1X,I2,16F8.0)
2919            GO TO 800
2920      500 IF (H.GE.0.0) GO TO 510
2921            IF (H.LT.-0.015) GO TO 505
2922    C       IF -15 METER LE H LT 0 , H IS SET TO 0
2923            H = 0.
2924            GO TO 510
2925    C       NO MORE COMPUTATIONS TO BE MADE IF HEIGHT LT -5 M
2926      505 NMORE = 0
2927            RETURN
2928    C.....GENERATE GRID OF 4D PROFILES IF PREVIOUS HEIGHT GE 30
2929      510 IF (H1.GE.30..OR.LOOK.EQ.1) CALL GEN4D
2930    C510    IF (H1.GE.30..OR.LOOK.EQ.1) CALL USGRID
2931    C       LOWER HEIGHT INDEX
2932            IHA=INT(H)
2933    C       LOWER HEIGHT INDEX
2934            HA = IHA*1.
2935            IWSX = IWSYM
2936    C       UPPER HEIGHT INDEX
2937            IHB = IHA + 1
2938            IF(IHB.LE.25) GO TO 513
2939            IHA=24
2940            HA=24.
2941            IHB=25
2942    C        UPPER HEIGHT
2943      513 HB = IHB*1.
2944    C.....LAT-LON INTERPOLATION OF 4D VALUES AT UPPER HEIGHT
2945      515 CALL INTER4(        PHI,THET,IHB,    P4D,D4D,T4D,PB,DB,TB,
2946          $ DPX4B,DPY4B,DTX4B,DTY4B,DPXXB,DPYYB,DPXYB)
2947            IF(IHA.EQ.0.AND.PB*DB*TB.LE.0.)GO TO 520
2948            GO TO 540
2949      520 IHB=IHB+1
2950    C.....LOOP TO FIND LOWEST VALID HEIGHT
2951            HB=HB+1.
2952            GO TO 515
2953      540 IF(IHA.GT.0) CALL INTER4(        PHI,THET,IHA,    P4D,D4D,T4D,
2954          1PA,DA,TA,DPX4A,DPY4A,DTX4A,DTY4A,DPXXA,DPYYA,DPXYA)
2955            IF (IWSYM .EQ.IAST) IWSX = IWSYM
2956            IF(IHA.EQ.0.OR.(PA*DA*TA.LE.0.AND.IHA.LT.10.AND.PB*DB*TB.GT.0.))
```

```
2957         1GO TO 550
2958          GO TO 600
2959   C.....LAT-LON INTERPOLATION OF 4D VALUES AT LOWER HEIGHT
2960      550 CALL INTER4(            PHI,THET,0,    P4D,D4D,T4D,
2961          .PA,DA,TA,DPX4A,DPY4A,DTX4A,DTY4A,DPXXA,DPYYA,DPXYA)
2962          IF (IWSYM .EQ. IAST) IWSX = IWSYM
2963          IF(TA-TB)560,570,560
2964      560 IF(TA*TB.LE.0.0)GO TO 570
2965          TZ=(TA-TB)/ALOG(TA/TB)
2966          GO TO 575
2967      570 TZ=TA
2968   C ...COMPUTES HEIGHT OF SURFACE
2969      575 HA=HB
2970          IF(PB*PA.LE.0.0)GO TO 576
2971          HA=HB+0.28705*TZ*ALOG(PB/PA)/G
2972      576 IF(H.GT.HA-0.04)GO TO 600
2973          PH=0.
2974          DH=0.
2975          TH=0.
2976          PGH=0.
2977          DGH=0.
2978          TGH=0.
2979          GO TO 800
2980   C.....HEIGHT INTERPOLATION OF P,D,T
2981      600 CALL INTER2(PA,DA,TA,HA,PB,DB,TB,HB,PGH,DGH,TGH,H)
2982   C.....HEIGHT INTERPOLATION OF DP/DX AND DP/DY
2983          CALL INTERW(DPX4A,DPY4A,HA,DPX4B,DPY4B,HB,DPX,DPY,H)
2984   C.....HEIGHT INTERPOLATION OF DT/DX AND DT/DY
2985          CALL INTERW(DTX4A,DTY4A,HA,DTX4B,DTY4B,HB,DTX,DTY,H)
2986   C.....HEIGHT INTERPOLATION OF D2P/DX2, D2P/DY2, AND D2P/DXDY
2987          CALL INTERZ(DPXXA,DPYYA,DPXYA,HA,DPXXB,DPYYB,DPXYB,HB,DPXX,DPYY,
2988         $DPXY,H)
2989   C       CHANGE OF NOTATION FOR OUTPUT
2990          PH = PGH
2991          DH = DGH
2992          TH = TGH
2993          IF(PH*DH*TH.LE.0.) GO TO 800
2994          CALL WIND
2995   C       CHANGE OF NOTATION FOR OUTPUT
2996          UH = UGH
2997          VH = VGH
2998          CP = 7.*PGH/(2.*DGH*TGH)
2999          DTZ = (TB - TA)/(1000.*(HB - HA))
3000   C.... VERTICAL MEAN WIND
3001          WGH = -CP*(UGH*DTX/DX5 + VGH*DTY/DY5)/(G+CP*DTZ+UH*DUH+VH*DVH)
3002   C       QBO=0 IF H LT 10
3003          IF (H.LT.10.) GO TO 800
3004          IF (IOPQ.EQ.2) GO TO 650
3005   C       COMPUTES QUASI BIENNIAL PERTURBATIONS
3006          CALL QBOGEN
3007   C       ADDS QBO PERTURBATIONS TO P,D,T
3008      650 PH=PGH*(1.+PQ)
3009          DH=DGH*(1.+DQ)
3010          TH=TGH*(1.+TQ)
3011   C       ADDS QBO WIND PERTURBATIONS TO U,V
3012          UH=UGH+UQ
3013          VH=VGH+VQ
3014   C.....THE FOLLOWING IS THE RANDOM PERTURBATIONS SECTION
3015   C.....NO RANDOM PERTURBATIONS IF IOPP GT 1
3016      800 CONTINUE
3017          IF(H.GT.30) GO TO 512
3018          IF (IPPT .NE. 0 .OR. IWSYM .NE. IAST) GO TO 512
3019   C::    WRITE(6,2000) (GLAT(I),I=1,16)
3020   C::    WRITE(6,2001) (GLON(I),I=1,16)
```

261

```
3021      C::   DO 504 I=1,26
3022      C::   IH=I-1
3023      C::   WRITE(6,2002) IH,(P4D(J,I),J=1,16)
3024      C::   504 CONTINUE
3025            IPRT=IPRT+1
3026      512 CONTINUE
3027            IF (IOPR.GT.1) GO TO 830
3028            IF(NPOP.EQ.0)GO TO 840
3029      C.....INTERPOLATES RANDOM WIND MAGNITUDES TO HEIGHT H, LATITUDE PHI
3030            CALL INTRUV(UR,VR,H,PHI,SUH,SVH)
3031            CALL INTRUV(PLP,DLP,H,PHI,PLPH,DLPH)
3032            CALL INTRUV(TLP,DLP,H,PHI,TLPH,DLPH)
3033            CALL INTRUV(ULP,VLP,H,PHI,ULPH,VLPH)
3034            CALL INTRUV(UDL,VDL,H,PHI,UDL2,VDL2)
3035            CALL INTRUV(UDS,VDS,H,PHI,UDS2,VDS2)
3036            SUHL=SQRT(ULPH*ABS(SUH))
3037            SUHS=SQRT((1.-ULPH)*ABS(SUH))
3038            SVHL=SQRT(VLPH*ABS(SVH))
3039            SVHS=SQRT((1.-VLPH)*ABS(SVH))
3040            SUH = SQRT(ABS(SUH))
3041            SVH = SQRT(ABS(SVH))
3042            IF(H.GE.25.)GO TO 805
3043      C.....IF H LE 20 USE 4D DATA RANDOM P,D,T SIGMAS
3044            IF (H.LE.20.) GO TO 810
3045      C.....INTERPOLATE PR,DR,TR ARRAYS TO GET P,D,T SIGMAS AT HEIGHT H,
3046      C           LATITUDE PHI
3047            CALL RTERP(25.,PHI,PR,DR,TR,SPHG,SDHG,STHG)
3048            GO TO 810
3049      805 CONTINUE
3050            CALL RTERP(H,PHI,PR,DR,TR,SPH,SDH,STH)
3051            GO TO 820
3052      C.....LAT-LON INTERPOLATION ON P,D,T SIGMAS AT LOWER HEIGHT
3053      810 CALL INTER4(          PHI,THET,IHA,    SP4,SD4,ST4,PA,DA,TA,
3054            $ DPX,DPY,DTX,DTY,DPXX,DPYY,DPXY)
3055      C.....LAT-LON INTERPOLATION ON P,D,T SIGMAS AT UPPER HEIGHT
3056            CALL INTER4(          PHI,THET,IHB,    SP4,SD4,ST4,PB,DB,TB,
3057            $ DPX,DPY,DTX,DTY,DPXX,DPYY,DPXY)
3058      C.....HEIGHT INTERPOLATION OF SIGMAS
3059            CALL INTERZ(PA,DA,TA,    HA,PB,DB,TB,    HB,SPH,SDH,STH,H)
3060            IF(SPH.LE.0.0.OR.SDH.LE.0.0.OR.STH.LE.0.0)GO TO 825
3061            IF(PH.LE.0.0.OR.DH.LE.0.0.OR.TH.LE.0.0)GO TO 825
3062            IF(H.LE.20.)GO TO 820
3063            FH = 1. - 0.2*(25. - H)
3064            SPH = FH*SPHG + (1. - FH)*SPH
3065            SDH = FH*SDHG + (1. - FH)*SDH
3066            STH = FH*STHG + (1. - FH)*STH
3067      C.....HEIGHT DISPLACEMENT BETWEEN PREVIOUS AND CURRENT POSITION
3068      820 DZ = H1 - H
3069            SPHL=SQRT(PLPH*ABS(SPH))
3070            SPHS=SQRT((1.-PLPH)*ABS(SPH))
3071            SDHL=SQRT(DLPH*ABS(SDH))
3072            SDHS=SQRT((1.-DLPH)*ABS(SDH))
3073            STHL=SQRT(TLPH*ABS(STH))
3074            STHS=SQRT((1.-TLPH)*ABS(STH))
3075            SPH = SQRT(ABS(SPH))
3076            SDH = SQRT(ABS(SDH))
3077            STH = SQRT(ABS(STH))
3078      C.....COMPUTES HORIZONTAL DISPLACEMENT DX BETWEEN PREVIOUS AND CURRENT
3079      C           POSITION, HORIZONTAL SCALE HL, AND VERTICAL SCALE VL
3080      C.....COMPUTES PERTURBATION VALUES PPH,DPH,TPH,UPH AND VPH
3081            CALL PERTRB
3082      C     ADDS RANDOM PERTURBATIONS TO PH,DH,TH
3083            PH = PH*(1. + PPP)
3084            DH = DH*(1. + DPH)
```

```
3085            TH = TH*(1. + TRH)
3086   C        ADDS RANDOM WINDS TO UH,VH
3087            UH=UH+URH
3088            VH=VH+VRH
3089   C.....SETS PREVIOUS RANDOM PERTURBATION IN P,D,T TO CURRENT
3090   C          PERTURBATIONS, FOR NEXT CYCLE
3091     825 RP1S= PRHS
3092            RD1S= DRHS
3093            RT1S= TRHS
3094            RP1L=PRHL
3095            RD1L=DRHL
3096            RT1L=TRHL
3097   C.....SETS PREVIOUS MAGNITUDES FO CURRENT VALUES, FOR NEXT CYCLE
3098            SP1S=SPHS
3099            SD1S= SDHS
3100            ST1S=STHS
3101            SP1L=SPHL
3102            SD1L=SDHL
3103            ST1L=STHL
3104   C.....SETS PREVIOUS WIND PERTURBATION VALUES TO CURRENT VALUES,
3105   C          FOR NEXT CYCLE
3106            RU1S=URHS
3107            RV1S=VRHS
3108            RU1L=URHL
3109            RV1L=VRHL
3110   C.....SETS PREVIOUS WIND PERTURBATION MAGNITUDES TO CURRENT VALUES,
3111   C          FOR NEXT CYCLE
3112            SU1S=SUHS
3113            SV1S=SVHS
3114            SU1L=SUHL
3115            SV1L=SVHL
3116   C.....SETS PREVIOUS HEIGHT TO CURRENT HEIGHT, FOR NEXT CYCLE
3117     830 H1 = H
3118   C.....SETS PREVIOUS LATITUDE TO CURRENT LATITUDE, FOR NEXT CYCLE
3119            PHI1R=PHIR
3120   C.....SETS PREVIOUS LONGITUDE TO CURRENT LONGITUDE, FOR NEXT CYCLE
3121            THET1R=THETR
3122   C        SETS NMORE TO COMPUTE MORE DATA ON NEXT CYCLE
3123     840 NMORE = 1
3124   C.....NO MORE DATA IF P, D, OR T LEQ 0
3125            IF(PH*DH*TH.LE.0.) RETURN
3126            CALL STDATM(H,TS,PS,DS)
3127            IF ((PS*DS*TS).GT.0.) GO TO 870
3128            PGHP=0.
3129            DGHP=0.
3130            TGHP=0.
3131            PHP=0.
3132            DHP=0.
3133            THP=0.
3134            GO TO 880
3135     870 PGHP=100.*(PGH-PS)/PS
3136            DGHP=100.*(DGH-DS)/DS
3137            TGHP=100.*(TGH-TS)/TS
3138            PHP=100.*(PH-PS)/PS
3139            DHP=100.*(DH-DS)/DS
3140            THP=100.*(TH-TS)/TS
3141   C        CONVERTS QBO RESULT TO PERCENT
3142     880 PQ=100.*PQ
3143            DQ=100.*DQ
3144            TQ=100.*TQ
3145   C        CONVERTS RANDOM RESULT TO PERCENT
3146            PRH=100.*PRH
3147            DRH=100.*DRH
3148            TRH=100.*TRH
```

```
3149          PRHS=100.*PRHS
3150          DRHS=100.*DRHS
3151          TRHS=100.*TRHS
3152          PRHL=100.*PRHL
3153          DRHL=100.*DRHL
3154          TRHL=100.*TRHL
3155          SPHS = 100.*SPHS
3156          SDHS = 100.*SDHS
3157          STHS = 100.*STHS
3158          SPHL = 100.*SPHL
3159          SDHL = 100.*SDHL
3160          STHL = 100.*STHL
3161    C     CONVERTS WIND SHEAR TO M/S/KM
3162          DUH = DUH * 1000.
3163          DVH = DVH * 1000.
3164    C      CONVERTS VERTICAL WIND TO CM/S
3165          WGH = WGH*100.
3166          PQA=PQA*100.
3167          DQA=DQA*100.
3168          TQA=TQA*100.
3169          SPH=SPH*100.
317           SDH=SDH*100.
3171          STH=STH*100.
3172          PSH=PSH*100.
3173          DSH=DSH*100.
3174          TSH=TSH*100.
3175          IF(NPOP.EQ.0) THEN
3176             UPRE=UGH
3177             VPRE=VGH
3178             DUPRE=DUH/1000.
3179             DVPRE=DVH/1000.
3180          ENDIF
3181    C
3182          RETURN
3183           ND
3184          SUBROUTINE SETUP
3185    C
3186    C.... SETUP HAS BEEN MODIFIED TO READ ALL INPUT FROM FILES WHICH
3187    C     ARE OPENED THEN CLOSED TO MINIMIZE I/O BUFFER SPACE REQUIRED
3188    C     BY THE PROGRAM.  AS SUCH, ALL PREDEFINED LUN'S ARE IGNORED.
3189    C
3190    C.... THE ORIGINAL SETUP ROUTINE ZEROED THE RANDOM PERTURBATIONS IF
3191    C     IOPR=2 AND ZEROED THE QUASI-BIENNIAL OSCILLATIONS IF IOPQ=2.
3192    C     THIS VERSION OF THE PROGRAM READS IN THE DATA REGARDLESS, THEN
3193    C     DISABLES THE RP'S AND THE QBO'S WHRE THEIR EFFECT IS SUMMED IN.
3194    C     THIS ALLOWS REAL-TIME CONTROL OF THESE FEATURES.
3195    C
3196    C.... WRITTEN   26 JAN 89   L SCHILLING   NASA/ADFRF.
3197    C
3198          CHARACTER*12 FILNAM
3199          DIMENSION IE(5),ID(5),IT(5),IDAY(12)
3200          DIMENSION IDTM(
3201          DIMENSION NDATA(  ,IX(10)
3202    C
3203          COMMON /INTEMP/  TEMP,IOTEM,,IV
3204                           ,NSAMP ,PF                    
3205                                                          
3206                              ,H     ,PHTIR ,THETIR,QMS(21),
3207                                                            
3208                                                            
3209                                                            
3210          COMMON /PCTTM/
3211                           IY4    MONTH ,IOPR  ,PG(18,19),TG(18,19),
3212                           
```

```
3213    .                    PAQ(17,5),DAQ(17,5),TAQ(17,5),PDQ(17,5),
3214    .                    DDQ(17,5),TDQ(17,5),PR(20,10),DR(20,10),
3215    .                    TR(20,10),UAQ(17,5),VAQ(17,5),UDQ(17,5),
3216    .                    VDQ(17,5),UR(25,10),VR(25,10),PQ      ,DQ      ,
3217    .                    TQ      ,UQ      ,VQ      ,PQA     ,DQA     ,TQA     ,UA      ,
3218    .                    VA      ,IOPQ    ,PLP(25,10),DLP(25,10),TLP(25,10),
3219    .                    ULP(25,10),VLP(25,10),UDL(25,10),VDL(25,10),
3220    .                    UDS(25,10),VDS(25,10)
3221           COMMON /CHIC  / DUM(18),IWSYM,UCOEF(14,9),VCOEF(14,9)
3222           DATA IDAY /  0, 31, 59, 90,120,151,181,212,243,273,304,334/
3223    C
3224           XMJD = 0.
3225           IF (MN.GT.12) GO TO 2
3226           IDA = IDAY(MN) + IDD
3227           DD = IDA
3228           IF (MOD(IYR,4).EQ.0.AND.MN.GT.2) IDA = IDA + 1
3229           XMJD = 2439856.0 + 365.0*(IYR-68.) + IDA + INT((IYR-65.0)/4.0)
3230    C
3231    C.... SECOND DATA CARD READS. FREE FIELD, LOGICAL UNIT NUMBERS FOR
3232    C     THE GRAM PROGRAM.  THESE LUNS ARE IGNORED IN THE OPEN FILE,
3233    C     READ FILE, CLOSE FILE APPROACH.  PROGRAM OPTIONS ARE ALSO READ
3234    C     IN.  THEY ARE DEFINE AS FOLLOWS:
3235    C
3236    C        IOPR=1:  RANDOM OUTPUT, =2:  NO RANDOM OUTPUT
3237    C        IOPQ=1:  QBO OUTPUT   , =2:  NO QBO OUTPUT
3238    C        NR1 = STARTING RANDOM NUMBER
3239    C
3240       2 READ(55,*) IUN    ,IUG    ,IUR    ,IUVC   ,IUQ    ,IUS    ,
3241         .              IU4    ,IOPR   ,IOPQ   ,NR1    ,IOTEM1,IOTEM2
3242    C
3243           IF (IOPR.LT.1.OR.IOPR.GT.2) GO TO 666
3244           IF (IOPQ.LT.1.OR.IOPQ.GT.2) GO TO 666
3245    C
3246           MONTH=MN
3247           RPSCALE = 1.0
3248    C:     R = RAND(NR1)          ! INIT CALLS MADE IN 'GRAMRT' IN
3249    C:     R = RAND(0)            ! REAL-TIME VERSION.  VALUE OF 1.0
3250    C:     R = RAND(0)            ! IS ASSUMED FOR NR1 SINCE NOT IN COMMON
3251    C
3252    C.... THIRD DATA CARD READS FREE FIELD, THE FOLLOWING DATA:
3253    C        RP1 = INITIAL RANDOM PRESSURE PERTURBATIONS, PERCENT
3254    C        RD1 = INITIAL RANDOM DENSITY PERTURBATION, PERCENT
3255    C        RT1 = INITIAL RANDOM TEMPERATURE PERTURBATION, PERCENT
3256    C        SD1 = INITIAL STANDARD DEVIATION FOR RANDOM DENSITY
3257    C              PERTURBATION, PERCENT
3258    C        RU1 = INITIAL EASTWARD WIND PERTURBATION, M/S
3259    C        RV1 = INITIAL NORTHWARD WIND PERTURBATION, M/S
3260    C        SU1 = INITIAL STANDARD DEVIATION FOR RANDOM EASTWARD WIND, M/S
3261    C        SV1 = INITIAL STANDARD DEVIATION FOR RANDOM NORTHWARD WIND, M/S
3262    C
3263           READ(55,*) RP1L  ,RP1S  ,RD1L  ,RD1S  ,RT1L  ,RT1S  ,
3264         .              RU1L  ,RU1S  ,RV1L  ,RV1S  ,RPSCALE
3265           IF (RPSCALE.LT.1.0 .OR. RPSCALE.GT.2.0) RPSCALE=1.0
3266           RP1=RP1L+RP1S
3267           RD1=RD1S+RD1L
3268           RT1=RT1S+RT1L
3269           RU1=RU1L+RU1S
3270           RV1=RV1L+RV1S
3271    C
3272    C      CONTINUE
3273    C
3274           CALL GETNM
3275    C
3276           IF (MONTH.LT.1) GO TO 12
```

```
3277    C
3278    C.... MONTH=13 IS ANNUAL AVERAGE CASE
3279    C
3280          M1=13
3281          M2=13
3282          GO TO 13
3283    C
3284    C.... M1 IS FOR NORTHERN HEMISPHERE, M2 FOR SOUTHERN.  M2=M1+6
3285    C     UNLESS M1=M2=13.
3286    C
3287       12 M1=MONTH
3288          M2=MONTH + 6
3289    C
3290    C.... SOUTHERN HEMISPHERE DATA IS 6 MONTHS DISPLACED FOR GROVES,
3291    C         STATIONARY PERTURBATIONS, AND RANDOM PERTURBATIONS
3292    C
3293          IF (M2.GT.12) M2=M2 - 12
3294    C
3295    C.... READ GROVES PRESSURE DATA.  CONVERT TO REAL AND STORE IN ARRAY.
3296    C
3297       13 CONTINUE
3298    C
3299          CLOSE(25)
3300          FILNAM='NASPGROVES.F'
3301          OPEN(25,FILE=FILNAM,STATUS='OLD',FORM='FORMATTED',
3302         .     ERR=999,IOSTAT=IOS)
3303          REWIND(25)
3304    C
3305          DO 100 I=1,234
3306          READ(25,111) IC,MI,IH,IY,IEX
3307      111 FORMAT(A2,13I4)
3308          IF (IC.NE.'P') GO TO 666
3309          IF (MI.EQ. M1) GO TO 30
3310          IF (MI.EQ. M2) GO TO 40
3311          GO TO 100
3312       30 KS=1
3313          GO TO 50
3314       40 KS=-1
3315       50 IH=(IH-20)/5
3316          TENX=10.**IEX
3317          DO 60 J=1,10
3318          K=10+KS*(J-1)
3319       60 PG(IH,K) = IX(J)*TENX
3320      100 CONTINUE
3321    C
3322    C.... READ GROVES DENSITY DATA.  CONVERT TO REAL AND STORE IN ARRAY.
3323    C
3324          DO 200 I=1,234
3325          READ(25,111) IC,MI,IH,IX,IEX
3326          IF (IC.NE.'D') GO TO 666
3327          IF (MI.EQ. M1) GO TO 130
3328          IF (MI.EQ. M2) GO TO 140
3329          GO TO 200
3330      130 KS=1
3331          GO TO 150
3332      140 KS=-1
3333      150 IH=(IH-20)/5
3334          TENX=10.**IEX
3335          DO 160 J=1,10
3336          K=10+KS*(J-1)
3337      160 PG(IH,K) = IX(J)*TENX
3338      200 CONTINUE
3339    C
3340    C.... READ GROVES TEMPERATURE DATA.  CONVERT TO REAL AND STORE IN ARRAY.
```

```
3341    C
3342            DO 300 I=1,234
3343            READ(25,111) IC,MI,IH,IX,IEX
3344            IF (IC .NE. 'T') GO TO 666
3345            IF (MI.EQ.M1) GO TO 230
3346            IF (MI.EQ.M2) GO TO 240
3347            GO TO 300
3348        230 KS=1
3349            GO TO 250
3350        240 KS=-1
3351        250 IH=(IH-20)/5
3352            TENX=10.**IEX
3353            DO 260 J=1,10
3354            K=10+KS*(J-1)
3355        260 TG(IH,K) = IX(J)*TENX
3356        300 CONTINUE
3357    C
3358    C.... ANNUAL MEAN CASE - BOTH HEMISPHERES EQUAL.
3359    C
3360            IF (MONTH.LT.13) GO TO 308
3361    C
3362            DO 304 I=1,18
3363            DO 304 J=1,9
3364            J20=20-J
3365            PG(I,J)=PG(I,J20)
3366            DG(I,J)=DG(I,J20)
3367            TG(I,J)=TG(I,J20)
3368        304 CONTINUE
3369    C
3370    C.... READ STATIONALY PERTURBATIONS DATA.  CONVERT TO REAL AND STORE
3371    C       IN PSP, DSP, AND TSP ARRAYS.
3372    C
3373        308 DO 360 I=1,1248
3374            READ(25,112) NDATA(1),(NDATA(KK),KK=2,19)
3375        112 FORMAT(A2,18I4)
3376            IC=NDATA(1)
3377            MI=NDATA(2)
3378            IH=NDATA(3)
3379            LON=NDATA(4)
3380            DO 311 K=1,5
3381            IP(K)=NDATA(4+K)
3382            ID(K)=NDATA(9+K)
3383        311 IT(K)=NDATA(14+K)
3384            IF (IC .NE. 'S') GO TO 666
3385            IF (MI.EQ.M1) GO TO 320
3386            IF (MI.EQ.M2) GO TO 330
3387            GO TO 360
3388        320 KS=1
3389            GO TO 340
3390        330 KS=-1
3391        340 ISH=2+(IH-44)/9
3392            L=(LON+20)/30
3393            IF(IH.LT.52) ISH = (IH-20)/10
3394            IF (IH.GT.84) ISH=9
3395            DO 350 J=1,5
3396            K=5+KS*(J+(KS-1) )
3397            PSP(ISH,K,L) = IP J*1000.
3398            DSP(ISH,K,L) = ID J*1000.
3399        350 TSP(ISH,K,L) = IT J*1000.
3400        36 CONTINUE
3401    C
3402    C.... ANNUAL MEAN CASE - BOTH HEMISPHERES EQUAL.
3403    C
3404            IF (MONTH.LT.13) GO TO 360
```

```
3405              DO 364 I=1,8
3406              DO 364 K=1,12
3407              DO 364 J=1,5
3408              J10=11-J
3409              PSP(I,J,K)=PSP(I,J10,K)
3410              DSP(I,J,K)=DSP(I,J10,K)
3411              TSP(I,J,K)=TSP(I,J10,K)
3412          364 CONTINUE
3413    C
3414          369 CONTINUE
3415    C
3416    C.... READ RANDOM PERTURBATIONS.
3417    C
3418              CLOSE(25)
3419              FILNAM='NASPRRW.F    '
3420              OPEN(25,FILE=FILNAM,STATUS='OLD',FORM='FORMATTED',
3421             .     ERR=999,IOSTAT=IOS)
3422              REWIND(25)
3423    C
3424              DO 430 I=1,260
3425              READ (25,112) IC,MI,IH,IP,ID,IT
3426          385 IF (IC .NE. 'R' ) GO TO 666
3427              IF (MI.EQ.M1) GO TO 390
3428              IF (MI.EQ.M2) GO TO 400
3429              GO TO 430
3430          390 KS=1
3431              GO TO 410
3432          400 KS=-1
3433          410 IF (IH.LT.95) IHR=(IH-20)/5
3434              IF (IH.GE.95) IHR = 14 + (IH - 80) / 20
3435              DO 420 J=1,5
3436              K = 5 + KS * (J + (KS - 1) / 2)
3437              PR(IHR,K) =(IP(J)*RPSCALE/1000.)**2
3438              DR(IHR,K) =(ID(J)*RPSCALE/1000.)**2
3439          420 TR(IHR,K) =(IT(J)*RPSCALE/1000.)**2
3440          430 CONTINUE
3441    C
3442    C.... ANNUAL MEAN CASE - BOTH HEMISPHERES EQUAL.
3443    C
3444              IF (MONTH.LT.13) GO TO 460
3445    C
3446              DO 435 I=1,20
3447              DO 435 J=1,5
3448              J10=11-J
3449              PR(I,J)=PR(I,J10)
3450              DR(I,J)=DR(I,J10)
3451              TR(I,J)=TR(I,J10)
3452          435 CONTINUE
3453    C
3454    C.... READ RANDOM WIND STANDARD DEVIATIONS.
3455    C
3456          460 DO 490 I=1,325
3457              READ(25,111) IC,MI,IH,IP,ID
3458          467 IF (IC .NE. 'RW') GO TO 666
3459              IF (MI.EQ.M1) GO TO 470
3460              IF (MI.EQ.M2) GO TO 475
3461              GO TO 490
3462          470 KS=1
3463              GO TO 480
3464          475 KS=-1
3465          480 IF (IH.LT.95) IHP=1+IH/5
3466              IF (IH.GE.95) IHP=19+(IH-80)/20
3467              DO 485 J=1,5
3468              K=5+KS*(J+(KS-1)/2)
```

```
3469              UR(IBR,K)=(IP(J)*RPSCALE)**2
3470         485  VR(IBR,K)=(ID(J)*RPSCALE)**2
3471         490  CONTINUE
3472    C
3473    C.... ANNUAL MEAN CASE - BOTH HEMISPHERES EQUAL.
3474    C
3475              IF (MONTH.LT.13) GO TO 500
3476              DO 495 I=1,25
3477              DO 495 J=1,5
3478              J10=11-J
3479              UR(I,J)=UR(I,J10)
3480              VR(I,J)=VR(I,J10)
3481         495  CONTINUE
3482    C
3483         500  CONTINUE
3484    C
3485    C.... READ ANNUAL PRESSURE, DENSITY, AND TEMPERATURE PERCENTS.
3486    C
3487              CLOSE(25)
3488              FILNAM='NASPPPWCS.F '
3489    C
3490              OPEN(25,FILE=FILNAM,STATUS='OLD',FORM='FORMATTED',
3491          .       ERR=999,IOSTAT=IOS)
3492              REWIND(25)
3493    C
3494              DO 840 I=1,25
3495              READ(25,112) IC,MI,IH,IP,ID,IT
3496         820  IF(IH.GT.90) IH=70+(IH/4)
3497              IH=1+(IH/5)
3498              IF (IC .NE.'P' .OR. IH .NE. I) GO TO 666
3499              DO 830 J=1,5
3500              PLP(I,J+5)=IP(J)/1000.
3501              PLP(I,6-J)=IP(J)/1000.
3502              DLP(I,J+5)=ID(J)/1000.
3503              DLP(I,6-J)=ID(J)/1000.
3504              TLP(I,J+5)=IT(J)/1000.
3505         830  TLP(I,6-J)=IT(J)/1000.
3506         840  CONTINUE
3507    C
3508    C.... READ WIND ANNUAL PERCENTS.
3509    C
3510              DO 865 I=1,25
3511              READ(25,113) IC,MI,IH,IP,ID
3512         113  FORMAT(A2,12I5)
3513         855  IF(IH.GT.90) IH=70+(IH/4)
3514              IH=1+(IH/5)
3515              IF (I .NE. IH .OR. IC .NE. 'PW') GO TO 666
3516              DO 860 J=1,5
3517              ULP(I,J+5)=IP(J)/1000.
3518              ULP(I,6-J)=IP(J)/1000.
3519              VLP(I,J+5)=ID(J)/1000.
3520         860  VLP(I,6-J)=ID(J)/1000.
3521         865  CONTINUE
3522    C
3523    C.... READ SMALL SCALE VELOCITY PERTURBATION CORRELATIONS.
3524    C
3525              DO 888 I=1,25
3526              READ(25,113) IC,MI,IH,IP,ID
3527         888  IF(IH.GT.90) IH=70+(IH/4)
3528              IH=1+(IH/5)
3529              IF (IH .NE. I .OR. IC .NE. 'US') GO TO 666
3530              DO 885 J=1,5
3531              UDS(I,J+5)=(IP(J)/1000.)
3532              UDS(I,6-J)=(IP(J)/1000.)
```

269

```
3533          VDS(I,J+5)=(ID(J)/1000.)
3534      885 VDS(I,6-J)=(ID(J)/1000.)
3535      888 CONTINUE
3536   C
3537   C.... READ LARGE SCALE VELOCITY PERTURBATION CORRELATIONS.
3538   C
3539          DO 898 I=1,25
3540          READ(25,113) IC,MI,IH,IP,ID
3541      894 IF(IH.GT.90) IH= 70+(IH/4)
3542          IH=1+(IH/5)
3543          IF (IH .NE. I .OR. IC .NE. 'CL')GO TO 666
3544          DO 896 J=1,5
3545          UDL(I,J+5)=(IP(J)/1000.)
3546          UDL(I,6-J)=(IP(J)/1000.)
3547          VDL(I,J+5)=(ID(J)/1000.)
3548      896 VDL(I,6-J)=(ID(J)/1000.)
3549      898 CONTINUE
3550   C
3551   C.... READ QUASI-BIENNIAL OSCILLATIONS (PRESSURE AMPLITUDE AND
3552   C      PRESSURE PHASE - DAYS PAST JAN 0, 1966).
3553   C
3554          CLOSE(25)
3555          FILNAM='NASPQBO.F    '
3556          OPEN(25,FILE=FILNAM,STATUS='OLD',FORM='FORMATTED',
3557        .      ERR=999,IOSTAT=IOS)
3558          REWIND(25)
3559   C
3560          DO 530 I=1,16
3561          READ(25,111) IC,IH,IX
3562      527 IF (IC .NE. 'QP' ) GO TO 666
3563          IH = (IH-5)/5
3564          DO 530 J=1,5
3565          PAQ(IH,J) = IX(2*J-1)/1000.
3566      530 PDQ(IH,J) = IX(2*J)*1.
3567          DO 531 I = 1,5
3568          PAQ(1,I) = 0.
3569      531 CALL PHASE(PDQ(2,I),15.,PDQ(3,I),20.,PDQ(1,I),10.)
3570   C
3571   C.... READ QBO DENSITY AMPLITUDE AND PHASE.
3572   C
3573          DO 540 I=1,16
3574          READ(25,111) IC,IH,IX
3575      537 IF (IC .NE. 'QD') GO TO 666
3576          IH=(IH-5)/5
3577          DO 540 J=1,5
3578          DAQ(IH,J) = IX(2*J-1)/1000.
3579      540 DDQ(IH,J)=IX(2*J)*1.
3580          DO 541 I = 1,5
3581          DAQ(1,I) = 0.
3582      541 CALL PHASE(DDQ(2,I),15.,DDQ(3,I),20.,DDQ(1,I),10.)
3583   C
3584   C.... READ QBO TEMPERATURE AMPLITUDE AND PHASE.
3585   C
3586          DO 550 I=1,16
3587          READ(25,111) IC,IH,IX
3588      547 IF (IC .NE. 'QT'  GO TO 666
3589          IH = (IH- 5).5
3590          DO 550 J=1,5
3591          TAQ(IH,J) = IX(2*J-1) 1000.
3592      550 TDQ(IH,J) = IX(2*J)*1.
3593          DO 551 I = 1,5
3594          TAQ(1,I) = 0.
3595      551 CALL PHASE(TDQ(2,I),15.,TDQ(3,I),20.,TDQ(1,I),10.)
3596   C
```

```
3597    C.... READ EASTWARD QBO WIND AMPLITUDE AND PHASE.
3598    C
3599          DO 560 I=1,16
3600          READ(25,111) IC,IH,IX
3601      557 IF (IC .NE. 'QU') GO TO 666
3602          IH=(IH- 5)/5
3603          DO 560 J=1,5
3604          UAQ(IH,J) = IX(2 * J - 1) / 10.
3605      560 UDQ(IH,J)=IX(2*J)*1.
3606          DO 561 I = 1,5
3607          UAQ(1,I) = 0.
3608      561 CALL PHASE(UDQ(2,I),15.,UDQ(3,I),20.,UDQ(1,I),10.)
3609    C
3610    C.... READ NORTHWARD QBO WIND AMPLITUDE AND PHASE.
3611    C
3612          DO 570 I=1,16
3613          READ(25,111) IC,IH,IX
3614      567 IF (IC .NE. 'QV') GO TO 666
3615          IH=(IH- 5)/5
3616          DO 570 J=1,5
3617          VAQ(IH,J) = IX(2 * J - 1) / 10.
3618      570 VDQ(IH,J) = IX(2*J)*1.
3619          DO 571 I = 1,5
3620          VAQ(1,I) = 0.
3621      571 CALL PHASE(VDQ(2,I),15.,VDQ(3,I),20.,VDQ(1,I),10.)
3622    C
3623    C.... READ IN SPHERICAL HARMONICS COEFFICIENTS
3624    C
3625          CLOSE(25)
3626          FILNAM='NASPSF.F     '
3627          OPEN(25,FILE=FILNAM,STATUS='OLD',FORM='FORMATTED',
3628         .     ERR=999,IOSTAT=IOS)
3629          REWIND(25)
3630    C
3631          DO 615 IFR=1,MN
3632          DO 613 JFR=1,14
3633          READ(25 ,640) IF1,IF2,(IDUM(I),I=1,9)
3634      640 FORMAT(2X,11I6)
3635          DO 613 I=1,9
3636      613 UCOEF(JFR,I)=FLOAT(IDUM(I))/100.
3637          DO 612 JFR=1,14
3638          READ(25 ,640) IF1,IF2,(IDUM(I),I=1,9)
3639          DO 612 I=1,9
3640      612 VCOEF(JFR,I)=FLOAT(IDUM(I))/100.
3641      615 CONTINUE
3642    C
3643          CLOSE(25)
3644    C
3645      621 R=H1
3646          IF(H1.LT.25.) R=25.
3647          CALL RTERP(R ,PHI1,PR,DR,TR,SP1,SD1,ST1)
3648          CALL INTRUV(PLP,DLP,H1,PHI1,PLP1,DLP1)
3649          CALL INTRUV(TLP,DLP,H1,PHI1,TLP1,R)
3650    C
3651          SP1L=SQRT(PLP1*ABS(SP1))*100.
3652          SP1S=SQRT((1.-PLP1)*ABS(SP1))*100.
3653          SD1L=SQRT(DLP1*ABS(SD1))*100.
3654          SD1S=SQRT((1.-DLP1)*ABS(SD1))*100.
3655          ST1L=SQRT(TLP1*ABS(ST1))*100.
3656          ST1S=SQRT((1.-TLP1)*ABS(ST1))*100.
3657    C
3658          CALL INTRUV(UR,VR,H1,PHI1,SU1,SV1)
3659          CALL INTRUV(ULP,VLP,H1,PHI1,ULP1,VLP1)
3660    C
```

```
3661          SU1L=SQRT(ULP1*ABS(SU1))
3662          SU1S=SQRT((1.-ULP1)*ABS(SU1))
3663          SV1L=SQRT(VLP1*ABS(SV1))
3664          SV1S=SQRT((1.-VLP1)*ABS(SV1))
3665    C
3666          CALL INTRUV(UDL,VDL,H1,PHI1,UDL1,VDL1)
3667          CALL INTRUV(UDS,VDS,H1,PHI1,UDS1,VDS1)
3668    C
3669          UDL1=UDL1*100.
3670          VDL1=VDL1*100.
3671          UDS1=UDS1*100.
3672          VDS1=VDS1*100.
3673    C
3674          RP1L=RP1L/100.
3675          RD1L=RD1L/100.
3676          RT1L=RT1L/100.
3677          SP1L=SP1L/100.
3678          SD1L=SD1L/100.
3679          ST1L=ST1L/100.
3680          RP1S=RP1S/100.
3681          RD1S=RD1S/100.
3682          RT1S=RT1S/100.
3683          SP1S=SP1S/100.
3684          SD1S=SD1S/100.
3685          ST1S=ST1S/100.
3686          UDL1=UDL1/100.
3687          VDL1=VDL1/100.
3688          UDS1=UDS1/100.
3689          VDS1=VDS1/100.
3690          RETURN
3691    C
3692      666 WRITE(6,700) FILNAM
3693      700 FORMAT('  ERROR IN SETUP INPUT FROM ',A12)
3694          STOP
3695    C
3696      999 CONTINUE
3697    C
3698    C.... OPEN ERROR ENCOUNTERED.
3699    C
3700          WRITE(6,677) FILNAM,IOS
3701      677 FORMAT(' OPEN ERROR ON FILE ',A12,' STATUS = ',I3)
3702          STOP
3703    C
3704          END
3705          SUBROUTINE SPHERE(MN,IH,PHIR,THETR,US,VS)
3706          COMMON/CHIC/DUM(18),IWSYM,UCOEF(14,9),VCOEF(14,9)
3707          DIMENSION Z(9)
3708          COSPHI=COS(PHIR)
3709          CSTHET=COS(THETR)
3710          SINPHI=SIN(PHIR)
3711          SNTHET=SIN(THETR)
3712          Z(1)=1.
3713          Z(2)=SINPHI
3714          Z(3)=CSTHET*COSPHI
3715          Z(4)=SNTHET*COSPHI
3716          Z(5)=(3*(SINPHI**2-1)/2.
3717          Z(6)=CSTHET*(3*COSPHI*SINPHI)
3718          Z(7)=SNTHET*(3*COSPHI*SINPHI)
3719          Z(8)=(2*(CSTHET**2-1)*(3*(COSPHI)**2)
3720          Z(9)=(2*SNTHET*CSTHET)*(3*(COSPHI)**2)
3721        5 IH5=IH/5-4
3722          IFP=9
3723          IF(IH.GT.65)IFP=4
3724          US=0.
```

272

```
3725          VS=0.
3726          DO 10 I=1,IFR
3727          US=US+Z(I)*UCOEF(IH5,I)
3728          VS=VS+Z(I)*VCOEF(IH5,I)
3729    10    CONTINUE
3730          RETURN
3731          END
3732          SUBROUTINE STDATM(Z,T,P,D)
3733          DIMENSION ZS(49),TMS(49),WMS(49),PS(49)
3734          DATA (ZS(I),I=1,49)/0., 11.019, 20.063, 32.162, 47.35,
3735         * 51.413, 71.802, 86.000, 91., 94.,   97., 100., 103., 106.,
3736         * 108., 110., 112., 115., 120., 125., 130., 135., 140., 145.,
3737         * 150., 155., 160., 165., 170., 180., 190.,210.,230., 265., 300.,
3738         * 350., 400., 450., 500., 550., 600., 650.,700.,750.,800.,850.,
3739         * 900., 950.,1000./
3740          DATA (TMS(I),I=1,49)/288.15, 216.65,216.65, 228.65, 270.65,270.65,
3741         * 214.65, 186.95, 186.87,187.74,190.4,195.08,202.23,212.89,223.29,
3742         * 240.,264.,300.,360.,417.23,469.27,516.59,559.63,598.78,
3743         * 634.39,666.8,696.29,723.13,747.57,790.07,825.31,
3744         * 878.84,915.78,955.2,976.01,990.06,995.83,998.22,
3745         * 999.24,999.67,999.85,999.93,999.97,999.99,999.99,
3746         *1000.,1000.,1000.,1000./
3747          DATA (WMS(I),I=1,49)/28.9644, 28.9644, 28.9644, 28.9644, 28.9644,
3748         * 28.9644, 28.9644, 28.9522, 28.889,28.783,  28.62,28.395,28.104,
3749         *27.765,27.521,27.268,27.020,26.680,26.205,25.803,25.436,25.087,
3750         *24.749,24.422,24.103,23.792,23.488,23.192,22.902,
3751         *22.342,21.809,20.825,19.952,18.688,17.726,16.735,
3752         *15.984,15.247,14.330,13.092,11.505,9.718,7.998,
3753         *6.579,5.543,4.849,4.404,4.122,3.940/
3754          DATA (PS(I),I=1,49)/1013.25, 226.32, 54.7487, 8.68014, 1.10905,
3755         * .66938,   .039564, 3.7338E-3, 1.5381E-3,9.0560E-4,5.3571E-4,
3756         * 3.2011E-4,1.9742E-4,1.2454E-4,9.3188E-5,7.1042E-5,5.5547E-5,
3757         * 4.0096E-5,2.5382E-5,1.7354E-5,1.25054E-5,9.3568E-6,
3758         * 7.2028E-6,5.6691E-6,4.5422E-6,3.6930E-6,3.0395E-6,
3759         * 2.5278E-6,2.1210E-6,1.5271E-6,1.1266E-6,6.4756E-7,
3760         * 3.9276E-7,1.7874E-7,8.7704E-8,3.4498E-8,1.451PE-9,
3761         * 6.4468E-9,3.0236E-9,1.5137E-9,8.2130E-10,4.8465E-10,
3762         * 3.1908E-10,2.2599E-10,1.7036E-10,1.3415E-10,1.0873E-10,
3763         * 8.9816E-11,7.5138E-11/
3764          IF(Z.LT.0.) GO TO 81
3765          RO=6356.766
3766          GO=9.80665
3767          WMO=28.9644
3768          RS=8314.32
3769          ZM=Z*1000.
3770          ROM=RO*1000.
3771          IF(Z.GE.86.) GO TO 6
3772          DO 3 I=1,7
3773          IF(ZS(I).LE.Z.AND.Z.LT.ZS(I+1)) GO TO 5
3774    3     CONTINUE
3775    5     ZL=RO*ZS(I)/(RO+ZS(I))
3776          ZU=RO*ZS(I+1)/(RO+ZS(I+1))
3777          ZLM=ZL*1000.
3778          ZUM=ZU*1000.
3779          WM=WMO
3780          HT=(RO*Z)/(RO+Z)
3781          HM=HT*1000.
3782          G=(TMS(I+1)-TMS(I))/(ZU-ZL)
3783          GM=G*.001
3784          IF(G.LT.0..OR.G.GT.0.) GO TO 12
3785          P=PS(I)*EXP(-(GO*WMO*(HM-ZLM))/(RS*TMS(I)))*100.
3786          GO TO 13
3787    12    P=PS(I)*((TMS(I)/(TMS(I)+G*(HT-ZL)))**((GO*WMO)/(RS*GM)))*100.
3788    13    T=TMS(I)+G*(HT-ZL)
```

```
3789        GO TO 25
3790      6 DO 7 I=8,48
3791        IF(ZS(I).LE.Z.AND.Z.LT.ZS(I+1)) GO TO 8
3792      7 CONTINUE
3793        I=48
3794        IF(Z.LE.1000.)GO TO 8
3795     81 T=0.
3796        P=0.
3797        D=0.
3798        RETURN
3799      8 IF(I.NE.8)GO TO 31
3800        T=TMS(9)
3801        GO TO 39
3802     31 IF(I.LT.16.OR.I.GE.19)GO TO 32
3803        T=240.+12.0*(Z-110.0)
3804        GO TO 39
3805     32 IF(I.GE.19)GO TO 33
3806        T=263.1905-76.3232*SQRT(1.-((Z-91.)/19.9429)**2)
3807        GO TO 39
3808     33 XI=(Z-120.)*(RO+120.)/(PO+Z)
3809        T=1000.-640.*EXP(-0.01875*XI)
3810     39 J=I
3811        IF(I.EQ.48)J=I-1
3812        Z0=ZS(J)
3813        Z1=ZS(J+1)
3814        Z2=ZS(J+2)
3815        WMA=WMS(J)*(Z-Z1)*(Z-Z2)/((Z0-Z1)*(Z0-Z2))+WMS(J+1)*(Z-Z0)
3816       &*(Z-Z2)/((Z1-Z0)*(Z1-Z2))+WMS(J+2)*(Z-Z0)*(Z-Z1)/
3817       &((Z2-Z0)*(Z2-Z1))
3818        ALP0=ALOG(PS(J))
3819        ALP1=ALOG(PS(J+1))
3820        ALP2=ALOG(PS(J+2))
3821        ALPA=ALP0*(Z-Z1)*(Z-Z2)/((Z0-Z1)*(Z0-Z2))+ALP1*(Z-Z0)
3822       &*(Z-Z2)/((Z1-Z0)*(Z1-Z2))+ALP2*(Z-Z0)*(Z-Z1)/
3823       &((Z2-Z0)*(Z2-Z1))
3824        ALPB=ALPA
3825        WMB=WMA
3826        IF(I.EQ.8.OR.I.EQ.48)GO TO 24
3827        J=J-1
3828        Z0=ZS(J)
3829        Z1=ZS(J+1)
3830        Z2=ZS(J+2)
3831        ALP0=ALOG(PS(J))
3832        ALP1=ALOG(PS(J+1))
3833        ALP2=ALOG(PS(J+2))
3834        ALPB=ALP0*(Z-Z1)*(Z-Z2)/((Z0-Z1)*(Z0-Z2))+ALP1*(Z-Z0)
3835       &*(Z-Z2)/((Z1-Z0)*(Z1-Z2))+ALP2*(Z-Z0)*(Z-Z1)/
3836       &((Z2-Z0)*(Z2-Z1))
3837        WMB=WMS(J)*(Z-Z1)*(Z-Z2)/((Z0-Z1)*(Z0-Z2))+WMS(J+1)*(Z-Z0)
3838       &*(Z-Z2)/((Z1-Z0)*(Z1-Z2))+WMS(J+2)*(Z-Z0)*(Z-Z1)/
3839       &((Z2-Z0)*(Z2-Z1))
3840     24 P=100.*EXP((ALFA-ALPB)/2.)
3841        WM=(WMA+WMB)/2.
3842     25 D=(WM*P)/(RS*T)
3843     26 RETURN
3844        END
3845        SUBROUTINE TINF
3846        COMMON/IOTEMP/IOTEM1,IOTEM2,IUG,IUN,DD,XMJD,FHI1,FHI,
3847       .           NSAME,PP1,RD1,RT1,SP1,SD1,ST1,PU1,PV1,SU1,SV1,
3848       $ MN,IDA,IYP,EL,FHI1P,THET1P,G,PI,H,FHIP,THETP,F10,F10B,GI,
3849       $ IHR,MIN,NMOPE,DL,HL,VL,DZ,DUMMY2(25)
3850        COMMON/COMJAC/XLAT,XLONG,SDA,SHA,DY,R,TE,EM
3851  C
3852  C    SUROUTINE TINF CALCULATES THE EXOSPRERIC TEMPERATURE ACCORDING TO JA
```

```
3853    C       SAO NO. 313 ,197 .
3854    C
3855    C LIST
3856    C       F10 = SOLAR RADI  NOISE FLUX (XE-22 WATTS/M**2)
3857    C       F10B= 81-DAY AVE AGE F10
3858    C       GI  = GEOMAGNETI  ACTIVITY INDEX,AP
3859    C       LAT = GEOGRAPHIC LATITUDE AT PERIGEE  (IN RAD)
3860    C       SDA = SOLAR DECLINATION ANGLE         (IN RAD)
3861    C       SHA = SOLAR HOUR ANGLE
3862    C       DY  = D/Y (DAY NUMBER/TROPICAL YEAR)? 1
3863    C       R = 0.31  (DIURNAL FACTOR)
3864    C
3865    C  CONSTANTS -- C=SOLAR ACTIVITY VARIATION. BETA,ETC. = DIURNAL VARIATI
3866    C              D=GEOMAGNETIC VARIATION. E=SIMIANNUAL VARIATION.
3867    C
3868            C1 = 383.0
3869            C2 =   3.32
3870            C3 =   1.80
3871    C
3872            PI = 3.14159265
3873            CON = 0.01745329252
3874            BETA= -37.0*CON
3875            GAMMA= 43.0*CON
3876            P   =   6.0*CON
3877            XM  =   2.5
3878            XNN =   3.0
3879    C
3880            D1 =  28.0
3881            D2 =   0.03
3882            D3 =   1.0
3883            D4 = 100.0
3884            D5 =  -0.08
3885    C
3886            E1 =   2.41
3887            E2 =   0.349
3888            E3 =   0.206
3889            E4 = 360.*CON
3890            E5 = 226.5*CON
3891            E6 = 720.*CON
3892            E7 = 247.6*CON
3893            E8 =   0.1145
3894            E9 =   0.5
3895            E10= E4
3896            E11= 342.3*CON
3897            E12=   2.16
3898    C
3899    C SOLAR ACTIVITY VARIATION
3900    C
3901            TC = C1 + C2*F10B + C3*(F10 - F10B)
3902    C
3903    C DIURNAL VARIATION
3904    C
3905            ETA    = 0.5*ABS(XLAT - SDA)
3906            THETA  = 0.5*ABS(XLAT + SDA)
3907            TAU    = SHA + BETA + P*SIN(SHA + GAMMA)
3908            TPI=2*PI
3909            IF(TAU) 210,230,
3910     210 IF(TAU+PI) 220,250,250
3911     220 TAU=TAU+TPI
3912         GO TO 210
3913     230 IF(TAU-PI) 250,250,240
3914     240 TAU=TAU-TPI
3915         GO TO 230
3916     250 CONTINUE
```

275

```
3917            A1 =(SIN(THETA))**XM
3918            A2 =(COS(ETA))**XM
3919            A3 =(COS(TAU/2.))**XNN
3920            B1 = 1.0 + R*A1
3921            B2 =(A2-A1)/B1
3922            TV = B1*( 1. + R*B2*A3)
3923            TL = TC*TV
3924      C
3925      C GEOMAGNETIC VARIATION
3926      C
3927            TG = D3*GI + D4*(1-EXP(D5*GI))
3928      C
3929      C SEMIANNUAL VARIATION
3930      C
3931            G3 = 0.5*(1.0 + SIN(E10*DY +E11) )
3932            G3 = G3**E12
3933            TAU1 = DY + E8*(G3 - E9)
3934            G1 = E2 + E3*(SIN(E4*TAU1 + E5))
3935            G2 = SIN(E6*TAU1+ E7)
3936            TS = E1 + F10B*G1*G2
3937      C
3938      C EXOSPHERIC TEMPERATURE
3939      C
3940            TE = TL + TG + TS
3941            RETURN
3942            END
3943            SUBROUTINE TME
3944            COMMON/COMJAC/XLAT,XLONG,SDA,SHA,DY,R,T,EM
3945            COMMON/IOTEMP/IOTEM1,IOTEM2,IUG,IUN,DD,XMJD,PHI1,PHI,
3946          .           NSAME,RP1, RD1, RT1, SP1, SD1, ST1, RU1, RV1, SU1, SV1,
3947          $ MN, IDA, IYR, H1, PHI1R,THET1R,G,RI,H,PHIR,THETR,F10,F10B,AP,
3948          $ IHR,MIN,NMORE,DX,HL,VL,DZ,DUMMY2(25)
3949      C
3950      C LIST
3951      C INPUT
3952      C    MN=MONTH.  IDA=DAY. IYR=HEAR. HR = HOUR.  MIN = MINUTE
3953      C    XLAT = LATITUDE (INPUT-GEOCENTRIC LATITUDE.)
3954      C    XLONG= LONGITUDE(INPUT-GEOCENTRIC LONGITUDE. OUTPUT  -180 TO + 180)
3955      C OUTPUT
3956      C    SDA = SOLAR DECLINATION ANGLE (IN RAD)
3957      C    SHA = SOLAR HOUR ANGLE (IN RAD)
3958      C    DD  = DAY NUMBER FROM 1JAN.
3959      C    DY  = DD/TROPICAL YEAR
3960      C
3961      C
3962      C SET CONSTANTS
3963      C
3964            YEAR = 365.2422
3965            YR=IYR
3966        6 DY = DD/YEAR
3967       30 FMJD = XMJD - 2435839.
3968      C
3969      C  CO,PUTE GREENWICH MEAN TIME IN MINUTES GMT
3970      C
3971            XHR =IHR
3972            XMIN = MIN
3973            GMT = 60*XHR + XMIN
3974      C
3975      C  COMPUTE GREENWICH MEAN POSITION - GP (IN DEG)
3976      C
3977            XJ = (XMJD - 2415020.0)/(36525.0)
3978            A1=99.6909833
3979            A2 = 36000.76854
3980            A3 = 0.00038708
```

276

```
3981          A4 = 0.25068447
3982          GP = A1 + A2*XJ - A3*XJ*XJ + A4*GMT
3983          N = GP/360.
3984          XN = N
3985          GP = GP - XN*360.
3986    C
3987    C  COMPUTE RIGHT ASCENSION POINT - RAP (IN DEG)
3988    C
3989    C     1ST CONVERT GEOCENTRIC LONGITUDE TO DEG LONGITUDE - WEST NEG $ EAS
3990    C
3991          IFACT = XLONG/180.
3992          XFACT = IFACT
3993          XLONG = 360. * XFACT - XLONG
3994    C
3995          RAP = GP + XLONG
3996          N = RAP/360.
3997          XN = N
3998          RAP = RAP - XN*360.
3999    C
4000    C  COMPUTE CELESTIAL LONGITUDE - XLS (IN RAD) - -PI/2 TO +PI/2
4001    C
4002          B1 = 0.017203
4003          B2 = 0.0335
4004          B3 = 1.410
4005          Y1 = B1*FMJD
4006          XLS = Y1 + B2*SIN(Y1) - B3
4007          TPI = 6.28318
4008          N = XLS/TPI
4009          XN = N
4010          XLS = XLS - XN*TPI
4011    C
4012    C  COMPUTE SOLAR DECLINATION ANGLE - SDA (IN RAD)
4013    C
4014          B4 = (TPI/360.)*23.45
4015          SDA = ASIN(SIN(DLS)*SIN(B4))
4016    C
4017    C  COMPUTE RIGHT ASCENSION OF SUN - RAS (IN RAD) - -PI/2 TO +PI/2
4018    C
4019          RAS = ASIN(TAN(SDA)/TAN(B4))
4020    C
4021    C  PUT RAS IN SAME QUADRANT AS XLS
4022    C
4023          PI = 3.14159265
4024          PI2 = PI/2.
4025          PI32= 3.*PI2
4026          RAS = ABS(RAS)
4027          TEMP = ABS(XLS)
4028          IF(TEMP - PI2) 130,130,100
4029      100 IF(TEMP - PI) 105,105,110
4030      105 RAS = PI - PAS
4031          GO TO 130
4032      110 IF(TEMP - PI32) 115,115,120
4033      115 PAS = PI + PAS
4034          GO TO 130
4035      120 PAS = TPI - PAS
4036      130 IF (XLS) 135,140,140
4037      135 PAS = -PAS
4038      140 CONTINUE
4039    C
4040    C  COMPUTE SOLAR HOUR ANGLE - SHA (IN DEG) - -
4041    C
4042          SHA = RAP*(PI/180.) - PAS
4043          IF(SHA) 210,230,230
4044      210 IF(SHA+PI) 220,250,250
```

277

```
4045        220 SHA=SHA+TPI
4046            GO TO 210
4047        230 IF(SHA-PI) 250,250,240
4048        240 SHA=SHA-TPI
4049            GO TO 230
4050        250 CONTINUE
4051    C
4052            RETURN
4053            END
4054            SUBROUTINE USGRID
4055    C
4056    C.... THIS ROUTINE GENERATES THE DATA FOR THE 16 POINT GRID USED BY
4057    C     THE GRAM PROGRAM AT ALTITUDES BELOW 25 KM (ALSO USED BETWEEN
4058    C     25 AND 30 KM FOR INTERPOLATION WITH THE GROVES MODEL).
4059    C
4060    C.... THE DATA CONSISTS OF PRESSURE, DENSITY, TEMPERATURE,
4061    C     PRESSURE VARIANCE, DENSITY VARIANCE, AND TEMPERATURE VARIANCE
4062    C     FOR LATITUDES 20-65 AND LONGITUDES 35-140 WEST (CONTINENTAL US +).
4063    C     DATA IS AT EACH 5 DEGREES OF LAT AND LONG, AND 0 TO 25 KM IN ONE
4064    C     KM INCREMENT.
4065    C
4066    C.... ALTHOUGH NOT THE MOST EFFICIENT, THIS ROUTINE IS DESIGNED TO
4067    C     INGERFACE WITH THE GRAM PROGRAM WITH MINIMUM OF IMPACT.  THUS THE
4068    C     NEW GRID POINTS ARE DETERMINED AS BEFORE (AS A FUNCTION OF WHERE
4069    C     YOU ARE AND WHERE YOU'RE HEADED).  THIS ROUTINE THEN OBTAINS THE
4070    C     DATA FROM MEMORY AND TRANSFERS IT TO THE ARRAY LOCATIONS EXPECTED
4071    C     BY THE GRAM PROGRAM.
4072    C
4073    C.... THIS ROUTINE ASSUMES THAT THE TRAJECTORY WHEN BELOW 25KM WILL
4074    C     ALWAYS LIE WITHIN LAT 20-65 AND LONG 35-140 (WEST).  OUTSIDE THIS
4075    C     AREA, DATA ON THE BORDER OF THE REGION WILL BE USED.  THUS THE
4076    C     TRAJECTORY IS ASSUMED TO ALWAYS LIE WITHIN THE NATIONAL
4077    C     METEOROLOGICAL CENTER DATA.  LOGIC FOR POLAR AND SOUTHERN
4078    C     HEMISPHERE DATA HAS BEEN REMOVED.
4079    C
4080    C.... WRITTEN   23 JAN 89   L SCHILLING   NASA/ADFRF.
4081    C
4082            COMMON /C4     /
4083            .               GLAT(16),GLON(16),NG,P(16,26),D(16,26),T(16,26),
4084            .               SP(16,26),SD(16,26),ST(16,26),PLON,CLON,HS
4085            COMMON /IOTEMP/ IOTEM1,IOTEM2,IUG   ,IUN   ,DDD   ,XMJD  ,PLAT  ,
4086            .               CLAT  ,NSAME ,RP1   ,RD1   ,RT1   ,SP1   ,SD1   ,
4087            .               ST1   ,RU1   ,RV1   ,SU1   ,SV1   ,MN    ,IDA   ,
4088            .               IYR   ,H1    ,PHI1R ,THET1R,GZ    ,RI    ,Z     ,
4089            .               PHIR  ,THETR ,F10   ,F10B  ,AP    ,IHR   ,MIN   ,
4090            .               NMORE ,DX    ,HL    ,VL    ,DZ    ,B     ,EPS   ,
4091            .               IOPP  ,LOOK  ,DUMMY(21)
4092            COMMON /PDTCOM/
4093            .               IU4   ,MONTH ,IOPR  ,PG(18,19),TG(18,19),
4094            .               DG(18,19),PSP(8,10,12),DSP(8,10,12),TSP(8,10,12),
4095            .               PAQ(17,5),DAQ(17,5),TAQ(17,5),PDQ(17,5),DDQ(17,5),
4096            .               TDQ(17,5),PR(20,10),DR(20,10),TR(20,10),UAQ(17,5),
4097            .               VAQ(17,5),UDQ(17,5),VDQ(17,5),UR(25,10),VR(25,10),
4098            .               PQ    ,DQ    ,TQ    ,UQ    ,VQ    ,PQA   ,DQA   ,
4099            .               TQA   ,UA    ,VA    ,IOPQ  ,PLP(25,10),DLP(25,10),
4100            .               TLP(25,10),ULP(25,10),VLP(25,10),UDL(25,10),
4101            .               VDL(25,10),UDS(25,10),VDS(25,10)
4102            COMMON /IPPTP/  IPPT
4103            COMMON /NASPGM/ PDAT(5720) ,DDAT(5720) ,TDAT(5720) ,
4104            .                   SPDAT(5720),SDDAT(5720),STDAT(5720)
4105    C
4106            IF(NSAME.EQ.1) RETURN
4107    C
4108            IPRT=0
```

```
4109          LOOK=0
4110          F = 0.017453293
4111          NG = 16
4112    C
4113    C.... LONG/LAT DISPLACEMENT FROM PREVIOUS TO CURRENT POSITION.
4114    C
4115          DX = PLON - CLON
4116          DY = CLAT - PLAT
4117    C
4118          IF (DY) 20,10,20
4119       10 IF (DX) 15,12,15
4120       12 K = 0
4121          GO TO 40
4122       15 THETA = 180. + SIGN(90.,DX)
4123          GO TO 30
4124       20 THETA = ATAN(DX/DY)/F
4125          IF (DY.GT.0.) THETA = THETA + 180.
4126          IF (THETA.LT.0.) THETA = THETA + 360.
4127    C
4128    C.... THETA = AZIMUTE ANGLE OF TRAJECTORY, USED TO ORIENT LAT-LON GRID
4129    C     COMPUTE INDEX USED IN COMPUTED GO TO FOR 110 THRU 180
4130    C
4131       30 K = INT((THETA + 67.5)/45.)
4132          IF (K.GT.8) K=K-8
4133    C
4134    C.... INITIAL ESTIMATE OF REFERNCE LATITUDE (LOWER LEFT GRID POINT)
4135    C
4136       40 LAT0 = 5*INT(CLAT/5.)
4137          IF (CLAT.LT.0.) LAT0 = LAT0 - 5
4138    C
4139    C.... INITIAL ESTIMATE OF REFERENCE LONGITUDE (LOWER LEFT GRID POINT)
4140    C
4141          LON0=5*INT(CLON/5.)
4142    C
4143    C.... ADJUSTS LAT0,LON0 ACCORDING TO DIRECTION OF TRAJECTORY AZIMUTH
4144    C
4145          IF (K.GT.0) GO TO 100
4146          LAT0 = LAT0 - 5
4147          LON0= LON0 + 10
4148          GO TO 190
4149      100 GO TO (110,120,130,140,150,160,170,180),K
4150      110 LAT0 = LAT0-10
4151          LON0 = LON0 + 10
4152          GO TO 190
4153      120 LAT0 = LAT0-10
4154          LON0 = LON0+15
4155          GO TO 190
4156      130 LAT0 = LAT0-5
4157          LON0 = LON0+15
4158          GO TO 190
4159      140 LON0 = LON0+15
4160          GO TO 190
4161      150 LON0 = LON0+10
4162          GO TO 190
4163      160 LON0 = LON0+5
4164          GO TO 190
4165      170 LAT0 = LAT0-5
4166          LON0 = LON0+5
4167          GO TO 190
4168      180 LAT0 = LAT0-10
4169          LON0 = LON0+5
4170      190 IF (LON0.GE.360) LON0 = LON0 - 360
4171          IF (LAT0.GT.75) LAT0 = 75
4172          DLI=1.25
```

```
4173          IF(ABS(CLAT).GE.18) GO TO 192
4174    C
4175          DLI=3.0
4176          LAT0=-18
4177    C
4178    C.... LATITUDE, LONGITUDE GRID AT 5 DEGREE INTERVALS
4179    C
4180      192 DO 195 I=1,4
4181          I12 = I+12
4182          DO 195 J=I,I12,4
4183          GLAT(J) = LAT0 + DLI*(J-I)
4184      195 GLON(J) = LON0 - 5. * (I - 1)
4185    C
4186    C.... PUT DATA FROM STORED US GRID IN ARRAYS.  THIS DATA HAS ALREADY
4187    C       BEEN TESTED FOR ZEROS, AND HAS GONE THROUGH ROUTINE 'ADJUST'
4188    C       PRIOR TO STORAGE, THUS THIS PROCESS HAS BEEN REMOVED FROM THIS
4189    C       ROUTINE.
4190    C
4191    C
4192    C.... TRANSFER DATA TO GRAM PROGRAM ARRAYS.
4193    C
4194          DO II=1,16
4195    C
4196    C....    COMPUTE SINGLY DIMENSIONED ARRAY INDEX.  LIMIT LATITUDE
4197    C        AND LONGITUDE COMPONENTS TO STAY WITHIN TABLE BOUNDARIES.
4198    C
4199          ILAT=GLAT(II)+0.1
4200          IF(ILAT.LT.20) ILAT=20
4201          IF(ILAT.GT.65) ILAT=65
4202    C
4203          ILON=GLON(II)+0.1
4204          IF(ILON.LT. 35) ILON= 35
4205          IF(ILON.GT.140) ILON=140
4206    C
4207          I1 = ((ILAT-20)/5)*572 + ((ILON-35)/5)*26 + 1
4208          I2 = I1 + 25
4209          I3 = 0
4210    C
4211          DO I=I1,I2
4212             I3 = I3+1
4213             P (II,I3) =  PDAT(I)
4214             D (II,I3) =  DDAT(I)
4215             T (II,I3) =  TDAT(I)
4216             SP(II,I3) = SPDAT(I)
4217             SD(II,I3) = SDDAT(I)
4218             ST(II,I3) = STDAT(I)
4219          ENDDO
4220    C
4221          ENDDO
4222    C
4223          RETURN
4224          END
4225          SUBROUTINE WIND
4226          COMMON /WINCOM/RHO,FCORY,DX5,DY5,PX,PY,PXX,PXY,PYY,U,V,   T,TX,TY,
4227        $   DU,DV,P,TPRE,TPRE,DUPRE,DVPRE
4228          COMMON /IOTEMP/DUM1(7),PHI,DUM2(11),MU,DM2A(5),G,R,H,PHIP,
4229        $THETR,DUM3(15),FLAT,DUMMY(19)
4230          COMMON/CHIC/DUM(19),IWSYM,UCOEF(14,9),VCOEF(14,9)
4231          ABSPHI=ABS(PHIP)
4232          IF (ABSPHI.LT.0.017453293*FLAT) GO TO 40
4233          IF (RHO.GT.0. .AND. T.GT.0. .AND. ABS(FCORY).GT.0.) GO TO 20
4234          U = 0.
4235          V = 0.
4236          DU = 0.0
```

```
4237          DV = 0.0
4238          IF (ABS(FCORY).LE.0.) GO TO 31
4239          RETURN
4240     20   FCORX = FCORY*DX5/DY5
4241          U = - PY/(FCORY*RHO)
4242          V =   PX/(FCORX*RHO)
4243          DU = -(G*TY)/(FCORY*T)
4244          DV =  (G*TX)/(FCORX*T)
4245     31   IF(H.GT.20.AND.H.LT.95.)GOTO 99
4246          IF (ABSPHI.GE.0.017453293*FLAT) RETURN
4247      40 CONTINUE
4248          U=UPRE
4249          V=VPRE
4250          DU=DUPRE
4251          DV=DVPRE
4252          IF (H.GT.20. .AND. H.LT.95.) GO TO 99
4253          RETURN
4254     C...SPHERICAL HARMONICS SECTION.................................
4255      99 IH=INT(H)
4256          IF(IH.LT.25)GOTO 130
4257          IF(IH.GE.90)GOTO 140
4258          IH1=5*INT(H/5.)
4259          IH2=IH1+5
4260          CALL SPHERE(MN,IH1,PHIR,THETR,US,VS)
4261          CALL SPHERE(MN,IH2,PHIR,THETR,US2,VS2)
4262          FACS=(H-IH1)/5.
4263          U=US+(US2-US)*FACS
4264          V=VS+(VS2-VS)*FACS
4265          DU=(US2-US)/5000.
4266          DV=(VS2-VS)/5000.
4267          RETURN
4268     C...LOW ALTITUDE FAIRING
4269      130 CALL SPHERE(MN,25,PHIR,THETR,US,VS)
4270          FACS=(H-20.)/5.
4271          FACG=1.-FACS
4272          U=FACG*U+FACS*US
4273          V=FACG*V+FACS*VS
4274          CALL SPHERE(MN,30,PHIR,THETR,US2,VS2)
4275          DUS=(US2-US)/5000.
4276          DVS=(VS2-VS)/5000.
4277          DU = FACG*DU + FACS*DUS
4278          DV = FACG*DV + FACS*DVS
4279          RETURN
4280     C...HIGH ALTITUDE FAIRING
4281      140 CALL SPHERE(MN,90,PHIR,THETR,US,VS)
4282          FACS=(H-90.)/5.
4283          FACG=1.-FACS
4284          U=FACS*U+FACG*US
4285          V=FACS*V+FACG*VS
4286          CALL SPHERE(MN,85,PHIR,THETR,US2,VS2)
4287          DUS=(US-US2)/5000.
4288          DVS=(VS-VS2)/5000.
4289          DU=FACG*DU+FACS*DUS
4290          DV=FACG*DV+FACS*DVS
4291          RETURN
4292          END
4293     .'T"2+
4294          FUNCTION PANF(X)                                      1094.600
4295     C                                                          1094.700
4296     C.... THIS FUNCTION IS A MODIFIED VERSION OF SUBROUTINE     1094.800
4297     C     RANDU. WHICH WAS WRITTEN FOR AN IBM 360.              1094.900
4298     C                                                          1095.000
4299     C.... DESCRIPTION OF VARIABLES:                            1095.100
4300     C          IX - FOR THE FIRST ENTRY THIS MUST CONTAIN ANY ODD INTEGER  1095.200
```

```
4301    C                  NUMBER WITH NINE OR LESS DIGITS.  AFTER THE FIRST ENTRY, 1095.300
4302    C                  IX WILL BE SET TO THE PREVIOUS VALUE OF IY, COMPUTED BY  1095.400
4303    C                  THIS ROUTINE.                                            1095.500
4304    C          IY  - A RESULTANT INTEGER RANDOM NUMBER REQUIRED FOR THE NEXT    1095.600
4305    C                  ENTRY TO THIS ROUTINE.  THE RANGE OF THIS NUMBER IS      1095.700
4306    C                  BETWEEN 0 AND 2**31                                      1095.800
4307    C          YFL - THE RESULTANT UNIFORM DISTRIBUTED, FLOATING POINT, RANDOM  1095.900
4308    C                  NUMBER IN THE RANGE 0.0 TO 1.0                           1096.000
4309    C          X   - DUMMY ARGUMENT.                                           1096.100
4310    C                                                                          1096.200
4311    C.... REMARKS:                                                             1096.300
4312    C          THIS ROUTINE IS SPECIFIC TO SYSTEM/360 AND WILL PRODUCE 2**29    1096.400
4313    C          TERMS BEFORE REPEATING.                                         1096.500
4314    C                                                                          1096.600
4315    C.... METHOD:                                                              1096.700
4316    C          POWER RESIDUE METHOD DISCUSSED IN IBM MANUAL C20-8011,           1096.800
4317    C          RANDOM NUMBER GENERATION AND TESTING.                           1096.900
4318    C                                                                          1097.000
4319          DATA IX /113/                                                        1097.100
4320    C                                                                          1097.200
4321          IY = IX*65539                                                        1097.300
4322    C                                                                          1097.400
4323          IF(IY) 5,6,6                                                         1097.500
4324    C                                                                          1097.600
4325        5 IY = IY + 2147483647 + 1                                             1097.700
4326    C                                                                          1097.800
4327        6 YFL = IY                                                             1097.900
4328    C                                                                          1098.000
4329          RANF = YFL*0.4656613E-9                                              1098.100
4330    C                                                                          1098.200
4331          IX = IY                                                              1098.300
4332    C                                                                          1098.400
4333          RETURN                                                               1098.500
4334          END                                                                  1098.600
4335    !!T72-
4336    C
4337          SUBROUTINE GRNGE ( RIO, RI, AE, BE, RES, IEMO, ICOORD, GRNGE )
4338    C
4339    C======================================================================C
4340    C                                                                      C
4341    C      PURPOSE    1) COMPUTE THE GROUND RANGE FROM POSITION (RIO) TO     C
4342    C                    POSITION (RI) IN THE APPROPRIATE COORDINATE SYSTEM  C
4343    C                    AS GIVEN BY (ICOORD) AND BY THE EARTH MODEL FLAG    C
4344    C                    (IEMO)                                             C
4345    C                                                                      C
4346    C      INPUTS     DESCRIPTION                                          C
4347    C                                                                      C
4348    C      (RIO)      MISSILE INITIAL POSITION VECTOR (RXI,RYI,RZI) (M)     C
4349    C      (RI)       MISSILE CURRENT POSITION VECTOR (RXI,RYI,RZI) (M)     C
4350    C      (AE)       EARTH MODEL SEMI-MAJOR AXIS (M)                       C
4351    C      (BE)       EARTH MODEL SEMI-MINOR AXIS (M)                       C
4352    C      (RES)      SPHERICAL EARTH RADIUS (M)                            C
4353    C      (IEMO)     EARTH MODEL FLAG:                                     C
4354    C                   = 0 - FLAT EARTH                                    C
4355    C                     1 - FLAT EARTH                                    C
4356    C                     2 - SPHERICAL EARTH                               C
4357    C                     3 - OBLATE EARTH                                  C
4358    C      (ICOORD)   COORDINATE SYSTEM IF EARTH-MODEL FLAG:               C
4359    C                   = 1 - FLAT EARTH OR SPHERICAL EARTH - AT SURFACE    C
4360    C                           X - LOCAL EAST                              C
4361    C                           Y - LOCAL NORTH                             C
4362    C                           Z - UP                                      C
4363    C                     2 - OBLATE OR SPHERICAL EARTH - EARTH CENTERED    C
4364    C                           X - B.R.P. MERIDIAN                         C
```

```
4365    C                          Y - NORMAL - EQUATORIAL PLANE              C
4366    C                          Z - THROUGH NORTH POLE                     C
4367    C                                                                     C
4368    C       OUTPUTS   DESCRIPTION                                         C
4369    C                                                                     C
4370    C       (GRNGE)   GROUND RANGE (ALONG THE SURFACE OF THE EARTH MODEL) C
4371    C                 FROM POSITION (RIO) TO POSITION (RI)                C
4372    C                                                                     C
4373    C   -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  C
4374    C                                                                     C
4375    C       REQUIREMENTS  1  SINGLE TO DOUBLE PRECISION       - DBLE      C
4376    C                     2  DOUBLE TO SINGLE PRECISION       - SNGL      C
4377    C                     3  D.P. SQUARE ROOT FUNCTION        - DSQRT     C
4378    C                     4  D.P. ARCTANGENT FUNCTION (+/- PI) - DATAN2   C
4379    C                     5  D.P. SINE FUNCTION               - DSIN      C
4380    C                     6  D.P. ARCCOSINE FUNCTION          - DACOS     C
4381    C                                                                     C
4382    C   -  -  -  -  -  -     -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  C
4383    C                                                                     C
4384    C       RESTRICTIONS 1) THE FLAGS FOR EARTH MODEL AND COORDINATE SYSTEM C
4385    C                       MUST CONFORM TO THE FOLLOWING:                C
4386    C                          IEMO    ICOORD                             C
4387    C                          ------  ------                             C
4388    C                            0       1                                C
4389    C                            1       1                                C
4390    C                            2       1                                C
4391    C                            2       2                                C
4392    C                            3       2                                C
4393    C                                                                     C
4394    C=====================================================================C
4395    C
4396            DIMENSION  RI(3    RIO(3)
4397    C
4398            DOUBLE PRECISION  DRXI, DRYI, DRZI, DRXE, DRYE, DRZE
4399            DOUBLE PRECISION  DXI2, DYI2, DZI2, DRE,  DECC
4400            DOUBLE PRECISION  DXE2, DYE2, DZE2, DAE,  DBE,  DTHE
4401            DOUBLE PRECISION  DRMI, DRME, DRR2, DTHR, DRES
4402            DOUBLE PRECISION  DSLA, DALG, DALA, DSL2, DRM
4403    C
4404            DOUBLE PRECISION  DF0,  DF1,  DF2
4405    C
4406            DATA  DF0, DF1, DF2 /  0.0D0, 1.D0, 2.0D0 /
4407    C
4408            IF( IEMO .GT. 1 ) GO TO 10
4409    C
4410    Cswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswsws
4411    C
4412    C....... FLAT EARTH      IEMO = 0 OR 1,  ICOORD = 1 (N/A)
4413    C               =========
4414    C
4415            GRNGE = SQRT( (RIO(1) - RI(1))**2 + (RIO(2) - RI(2))**2 )
4416    C
4417            GO TO 100
4418    C
4419    Cswswswswswswswswswsws  swswswswswswswswswswswswswswswswswswswswswsws
4420    C
4421     10 CONTINUE
4422    C
4423    C....... OTHER THAN FLAT EARTH
4424    C               ==========
4425    C
4426            DRXI = DBLE ( RI(1) )
4427            DRYI = DBLE ( RI(2) )
4428            DRZI = DBLE ( RI(3) )
```

```
4429    C
4430            DRXE = DBLE ( RIO(1) )
4431            DRYE = DBLE ( RIO(2) )
4432            DRZE = DBLE ( RIO(3) )
4433    C
4434            IF( IEMO .EQ. 3 ) GO TO 20
4435    C
4436    C....... SPHERICAL EARTH        IEMO = 2,   ICOORD = 1 OR 2
4437    C            ----------
4438    C
4439            DRES = DBLE ( RES )
4440    C
4441            IF( ICOORD .EQ. 1 ) DRZI = DRZI + DRES
4442            IF( ICOORD .EQ. 1 ) DRZE = DRZE + DRES
4443    C
4444            GO TO 40
4445    C
4446    Cswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswsws
4447    C
4448        20 CONTINUE
4449    C
4450    C....... OBLATE EARTH        IEMO = 3,   ICOORD = 2
4451    C            ------
4452    C
4453            DAE  = DBLE ( AE )
4454            DBE  = DBLE ( BE )
4455            DECC = (DAE/DBE)**2
4456    C
4457            DXI2 = DRXI**2
4458            DYI2 = DRYI**2
4459    C
4460            DALG = DSQRT ( DXI2  + DYI2 )
4461            DALA = DF0
4462            IF(( DRZI .NE. DF0 ).OR.
4463          *   ( DALG .NE. DF0 ))   DALA = DATAN2( DRZI, DALG )
4464    C
4465            DSLA  = DSIN( DALA )
4466            DSL2  = DSLA*DSLA
4467    C
4468            DRE   = DAE/DSQRT( DF1 + (DECC-DF1)*DSL2 )
4469    C
4470            DXE2 = DRXE**2
4471            DYE2 = DRYE**2
4472    C
4473            DALG = DSQRT ( DXE2  + DYE2 )
4474            DALA = DF0
4475            IF(( DRZE .NE. DF0 ).OR.
4476          *   ( DALG .NE. DF0 ))   DALA = DATAN2( DRZE, DALG )
4477    C
4478            DSLA = DSIN( DALA )
4479            DSL2 = DSLA*DSLA
4480    C
4481            DRES = ( DRE + DAE/DSQRT( DF1 + (DECC-DF1)*DSL2 ) )/DF2
4482    C
4483    Cswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswswsws
4484    C
4485        40 CONTINUE
4486    C
4487    C....... GROUND RANGE COMPUTATION
4488    C
4489            DPMI = DPXI*DPXI + DPYI*DPYI + DRZI*DPZI
4490            DPME = DRXE*DPXE + DRYE*DPYE + DRZE*DRZE
4491    C
4492            DTHR = DPXI*DPXE + DPYI*DRYE + DRZI*DRZE
```

```
4493        DRR2 = DSQRT (   -MI*DRME )
4494        DTHE = DF0
4495   C
4496        IF( DRR2 .NE.    0 ) DTHE = DTHR/DRR2
4497        IF( DTHE .GT.    1 ) DTHE =  DF1
4498        IF( DTHE .LT.   -1 ) DTHE = -DF1
4499   C
4500        DTHE = DACOS(   E )
4501        DRM  = DRES*DT
4502   C
4503        GRNGE = SNGL    M )
4504   C
4505   CSWSWSWSWSWSWSWSWSWS    SWSWSWSWSWSWSWSWSWSWSWSWSWSWSWSWSWSWS
4506   C
4507     100 CONTINUE
4508   C
4509        RETURN
4510        END
4511   !!T72+                                                           GETN   1
4512        SUBROUTINE GET                                              GETN   2
4513   C                                                                GETN   3
4514   C     READS  SETUP   ATA TAPE,  OR NMC GRID DATA CARDS,          GETN   4
4515   C     AND WRITES SCRA  H FILE FOR USE BY SELEC4.                 GETN   5
4516   C                                                                GETN   6
4517        DIMENSION IP(15                                            GETN   7
4518   C                                                                GETN   8
4519        COMMON /IOTEMP   IOTEM1,IOTEM2,IUG,IUN,IDUM(60)             GETN   9
4520   C
4521        OPEN(UNIT=IOTE   ,STATUS='SCRATCH',FORM='UNFORMATTED')
4522        NREC=0                                                     GETN  10
4523        IF(IUN.EQ.5) GO  O 2                                       GETN  11
4524   C                                                               GETN  12
4525        OPEN(UNIT=IUG, F LE='NMC.DAT',STATUS='OLD',READONLY)
4526   1    READ(IUN,300,EN =90) N,IP                                  GETN  13
4527   300  FORMAT(A2,19I7                                             GETN  14
4528        IF(N.NE.'N ')    TO 6                                      GETN  15
4529        GO TO 3                                                    GETN  16
4530   2    READ(5,100) IP                                            GETN  17
4531   100  FORMAT(15I5)                                              GETN  18
4532   3    DO 4 I=1,15,3                                             GETN  19
4533        M=IP(I)                                                   GETN  20
4534        IF(M.LT.1) GO T  5                                        GETN  21
4535        IJ=IP(I+1)*100   P(I+2)                                   GETN  22
4536        WRITE(IOTEM2) I                                           GETN  23
4537        NREC=NREC+1                                               GETN  24
4538   4    CONTINUE                                                  GETN  25
4539        IF(IUN.EQ.5) G  TO 2                                      GETN  26
4540        GO TO 1                                                   GETN  27
4541   5    IF(NREC.NE.19    GO TO 6                                  GETN  28
4542        IF(IUN.EQ.5) G   42
4543        IF(IUN.NE.IUG     TO 42
4544   C    MOVES PAST FIR   OF ON UNIT IUG                           GETN  29
4545   41   READ(IUG,9999,E  =42) IDUMMY                              GETN  30
4546   9999 FORMAT(A10)                                               GETN  31
4547        GO TO 41                                                  GETN  32
4548   42   RETURN                                                    GETN  33
4549   6    STOP                                                      GETN  36
4550   90   WRITE(6,400) I                                            GETN  37
4551   400  FORMAT (1H   FF  ATURE END-OF-FILE FOUND ON UNIT '.I2.
4552        S O CALLED FROM   EROUTINE GETNMC.')
4553        STOP                                                      GETN  4
4554        END                                                       GETN  41
4555   !!T72-
```

```
0001   c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003   !!G toolbox2.finc
0004
0005   c.....Load the ToolBox traps
0006
0007   !!M Inlines.f
0008
0009   c-------------------------------------------------------------------------------
0010         subroutine SaveMissionFile
0011   c-------------------------------------------------------------------------------
0012   c     Save a mission file's data and resource forks using a newly opened
0013   c     file (which may overwrite an existing file).  Then, close the file.
0014   c     (The dialog box contents have already been checked for validity.)
0015   c
0016
0017   c.....file info
0018
0019         include 'FileInfo.inc'
0020         include 'RunSetup.inc'
0021         include 'traj.inc'
0022         include 'TrjCom.inc'
0023
0024   c.....set up pointer for QuickDraw globals
0025
0026         pointer / QDGlobals /       qdg
0027         common  / QDGPtr /          qdg
0028
0029   c.....various character strings
0030
0031         string*255                  Prompt
0032         string*255                  MissionFileName
0033         string*255                  RezLabel
0034         string*255                  ItemText
0035         character*255               CharString
0036
0037   c.....reply record
0038
0039         record / SFReply    /       SFR
0040
0041   c.....prompt location
0042
0043         record / Point      /       PrmptPt
0044
0045   c.....cursor handle
0046
0047         record / CursHandle /       CursorHndl
0048
0049   c.....file information parameters
0050
0051         record /  SFTypeList / SFT
0052         record / FInfo       / fndrInfo
0053         character*4            FilTyp
0054         character*4            fMaker
0055         string*255             FilNam
0056
0057   c.....I/O error flags
0058
0059         integer*2               iosErr,       ios
0060
0061   c.....screen position info
0062
0063         integer*2 menuHeight
0064         integer*4 left, bottom, top, right
```

```
0065          integer*4 dialc .eight, dialogWidth
0066
0067   c.....dimensions of s' ndard SFPutFile dialog box (InsideMac, Chapter 47)
0068
0069          parameter          ( dialogHeight      - 136 )
0070          parameter          ( dialogWidth       - 348 )
0071
0072   c.....handle for STP  ta
0073
0074          record / String indle / STRHndl
0075          record / Handle       / RezHndl
0076
0077   c.....varia  a name l els for external plot file
0078
0079          character*16 Na  (7)
0080
0081   c--------------------  ------------------------------------------------------
0082
0083   c.....set the prompt
0084
0085          prompt = 'Save  ssion data file as'
0086
0087   c.....get the menuHe:  t (don't assume it is fixed at 20!)
0088
0089          menuHeight = G  BarHeight()
0090
0091   c.....get the screen   tents (use i*4 for screen math per Mac Tech Note 117)
0092
0093          left   = qdg^.s  eenBits.bounds.left
0094          right  = qdg^.s  eenBits.bounds.right
0095          bottom = qdg^.   eenBits.bounds.bottom
0096          top    = qdg^.s  eenBits.bounds.top + int4(menuHeight)
0097
0098   c.....set the left an  top edges of the save file dialog
0099
0100          PrmptPt.H =     right - left) - dialogWidth  ) / 2
0101          PrmptPt.V = (   bottom - top) - dialogHeight ) / 2 ) + menuHeight
0102
0103   c.....check to see if  e already have the filename for saving
0104
0105          if( .not. iGot  File ) then
0106            MissionFileNa  = 'BDPS Mission Data'
0107          endif
0108
0109   c.....get the target  lename for the save (put) operation
0110
0111          call SFPutFile  %val(PrmptPt), %val(Prompt), %val(MissionFileName),
0112       &                  %val(int4(nil)), %ref(SFR) )
0113
0114   c.....RETURN if no mi sion data file was selected (cancel)
0115
0116          if ( .not. SFR.  od ) then
0117            return
0118          else
0119
0120            MissionFileN   = SFR.fName
0121
0122            iosErr = Set     %val(int4(nil)) , %val(SFR.vRefNum) )
0123
0124   c........open the new   e fork
0125
0126            open( unit=1  file=MissionFileName, creator='MDoF', filetype='rtdf',
0127       &           status= ew', access='sequential' )
0128
```

287

```
0129    c.......open the ne    ource fork
0130
0131            call Create    le( %ref(SFR.fname) )
0132            ios    = Res    r
0133            RefNum = Ope   sFile ( %ref(SFR.fname) )
0134            ios    = Res.   r
0135
0136    c.......setup the mi    on label resource
0137
0138            STRHndl  = N    ndle( %val(int4(255) ) )
0139            RezLabel = 'M   sion Text'
0140            call AddReso   e( %val(STRHndl), %val('STR '),
0141         &                  %val(int2(rOldMissionText)), %ref(RezLabel) )
0142            RezHndl = STR   dl
0143            call WriteRes  rce( %val(RezHndl) )
0144
0145    c.......setup the lat  ude resource
0146
0147            STRHndl  = Ne  andle( %val(int4(255) ) )
0148            RezLabel = 'I   tial Latitude'
0149            call AddResou  e( %val(STRHndl), %val('STR '),
0150         &                  %val(int2(rOldLatitude)), %ref(RezLabel) )
0151            RezHndl = STR  dl
0152            call WriteRes  rce( %val(RezHndl) )
0153
0154    c.......setup the lon  tude resource
0155
0156            STRHndl  = Ne  andle( %val(int4(255) ) )
0157            RezLabel =  I  tial Longitude'
0158            call AddResou  e( %val(STRHndl), %val('STR '),
0159         &                  %val(int2(rOldLongitude)), %ref(RezLabel) )
0160            RezHndl = STRHndl
0161            call WriteResource( %val(RezHndl) )
0162
0163    c.......setup the duration resource
0164
0165            STRHndl  = NewHandle( %val(int4(255) ) )
0166            RezLabel = 'Flight Duration'
0167            call AddResource( %val(STRHndl), %val('STR '),
0168         &                  %val(int2(rOldDuration)), %ref(RezLabel) )
0169            RezHndl = STRHndl
0170            call WriteRes  rce( %val(RezHndl) )
0171
0172    c.......setup the altitude resource
0173
0174            STRHndl  = NewHandle( %val(int4(255) ) )
0175            RezLabel = 'Initial Altitude'
0176            call AddResource( %val(STRHndl), %val('STR '),
0177         &                  %val(int2(rOldAltitude)), %ref(RezLabel) )
0178            RezHndl = STRHndl
0179            call WriteResource( %val(RezHndl) )
0180
0181    c.......setup the ascent selection resource
0182
0183            STRHndl  = NewHandle( %val(int4(255)   )
0184            RezLabel = 'Ascent Profile'
0185            call AddResource  %val(STRHndl), %val('STR '),
0186         &                  %val(int2(rOldAscent)), %ref(RezLabel) )
0187            RezHndl = STRHndl
0188            call WriteReso  rce  %val(RezHndl)
0189
0190    c.......setup the climate selection resource
0191
0192            STRHndl  = NewHandle( %val(int4(255)   )
```

```
0193              RezLabel = 'Climate File'
0194              call AddResource( %val(STRHndl), %val('STR '),
0195        &                      %val(int2(rOldClimate)), %ref(RezLabel) )
0196              RezHndl = STRHndl
0197              call WriteResource( %val(RezHndl) )
0198
0199    c.......setup the deg West/deg East radio button resource
0200
0201              STRHndl  = NewHandle( %val(int4(255) ) )
0202              RezLabel = 'Degree Radio'
0203              call AddResource( %val(STRHndl), %val('STR '),
0204        &                      %val(int2(rOldDegRadio)), %ref(RezLabel) )
0205              RezHndl = STRHndl
0206              call WriteResource( %val(RezHndl) )
0207
0208    c.  ....setup the m/km radio button resource
0209              STRHndl  = NewHandle( %val(int4(255) ) )
0210              RezLabel = 'Distance Radio'
0211              call AddResource( %val(STRHndl), %val('STR '),
0212        &                      %val(int2(rOldDistRadio)), %ref(RezLabel) )
0213              RezHndl = STRHndl
0214              call WriteResource( %val(RezHndl) )
0215
0216    c.......setup the sec/min/hr radio button resource
0217
0218              STRHndl  = NewHandle( %val(int4(255) ) )
0219              RezLabel = 'Time Radio'
0220              call AddResource( %val(STRHndl), %val('STR '),
0221        &                      %val(int2(rOldTimeRadio)), %ref(RezLabel) )
0222              RezHndl = STRHndl
0223              call WriteResource( %val(RezHndl) )
0224
0225    c.......now set the flag because we have a target for saving
0226
0227              iGotOldFile = .true.
0228
0229          endif
0230
0231    c.....use watch cursor while writing data and resources
0232
0233              cursorHndl = GetCursor ( %val(int2(4)) )
0234              call SetCursor ( %val(cursorHndl.CRHDL^.CRPTR^) )
0235
0236    c.....at this point in the logic, the file's forks should both be open;
0237
0238    c.....for each item, we check to see if the open resource file already
0239    c.....contains that resource; if it's there, we change it; otherwise,
0240    c.....we add a resource to the open resource file.
0241
0242    c.....save the mission label resource
0243
0244          call GetDItem( %val(GetSelection), %val(rMissionLabel),
0245        &               %ref(DType), %ref(DItem), %ref(tempRect) )
0246          call GetIText ( %val(DItem) , %val(ItemText) )
0247          RezHndl = GetResource( %val('STR '), %val(int2(rOldMissionText)) )
0248          STRHndl = RezHndl
0249          STRHndl.shdl^..^ = ItemText
0250          call ChangedResource( %val(RezHndl) )
0251          call WriteResource ( %val(RezHndl) )
0252
0253    c.....save the latitude resource
0254
0255          call GetDItem( %val(GetSelection), %val(rLatitude),
0256        &               %ref(DType), %ref(DItem), %ref(tempRect) )
```

```
0257          call GetIText    val(DItem) , %val(ItemText) )
0258          RezHndl.bhdl =   tResource(%val('STR '), %val(int2(rOldLatitude)))
0259          STRHndl = RezE:
0260          STRHndl.shdl^.   r^ = ItemText
0261          call ChangedRe:  rce( %val(RezHndl) )
0262          call WriteReso:  e  ( %val(RezHndl) )
0263
0264    c.....save the longi:  e resource
0265
0266          call GetDItem    al(GetSelection), %val(rLongitude),
0267       &                   ef(DType), %ref(DItem), %ref(tempRect) )
0268          call GetIText    val(DItem) , %val(ItemText) )
0269          RezHndl.bhdl =   tResource(%val('STR '), %val(int2(rOldLongitude)))
0270          STRHndl = RezE:
0271          STRHndl.shdl^.s  r^ = ItemText
0272          call ChangedRe   rce( %val(RezHndl) )
0273          call WriteReso:  e  ( %val(RezHndl) )
0274
0275    c.....save the durat:  resource
0276
0277          call GetDItem(   al(GetSelection), %val(rDuration),
0278       &                   ef(DType), %ref(DItem), %ref(tempRect) )
0279          call GetIText '  val(DItem) , %val(ItemText) )
0280          RezHndl.bhdl =   tResource(%val('STR '), %val(int2(rOldDuration)))
0281          STRHndl = RezB:
0282          STRHndl.shdl^.s  r^ = ItemText
0283          call ChangedRes  rce( %val(RezHndl) )
0284          call WriteResou  e  ( %val(RezHndl) )
0285
0286    c.....save the altitu   resource
0287
0288          call GetDItem(   l(GetSelection), %val(rAltitude),
0289       &                   ef(DType), %ref(DItem), %ref(tempRect) )
0290          call GetIText :  val(DItem) , %val(ItemText) )
0291          RezHndl.bhdl =   tResource(%val('STR '), %val(int2(rOldAltitude)))
0292          STRHndl = RezHn
0293          STRHndl.shdl^.s  r^ = ItemText
0294          call ChangedRes  rce( %val(RezHndl) )
0295          call WriteResou  e  ( %val(RezHndl) )
0296
0297    c.....save the ascen:  election resource
0298
0299          ItemText    =    centSelection
0300          RezHndl.bhdl =   tResource(%val('STR '), %val(int2(rOldAscent)))
0301          STRHndl = RezE:
0302          STRHndl.shdl^.   r^ = ItemText
0303          call ChangedRes  rce( %val(RezHndl) )
0304          call WriteReso:  e  ( %val(RezHndl) )
0305
0306    c.....save the climat  selection resource
0307
0308          write(CharStri:  *) ClimateSelection
0309          ItemText = Cha:  ring
0310          RezHndl.bhdl =   tResource( %val('STR '), %val(int2(rOldClimate)))
0311          STRHndl = RezH
0312          STRHndl.shdl     =  ItemText
0313          call ChangedRe   rce  %val(RezHndl) :
0314          call WriteRes:   = / %val(RezHndl) )
0315
0316    c.....save the state   he deg West deg East radio button
0317
0318          write CharStri   * rDegreeSelection
0319          ItemText = Cha:  ring
0320          RezHndl.bhdl =   Resource( %val('STR '), %val(int2(rOldDegRadio)))
```

```
0321            STRHndl = RezHndl
0322            STRHndl.shdl^.sptr^ = ItemText
0323            call ChangedResource( %val(RezHndl) )
0324            call WriteResource  ( %val(RezHndl) )
0325
0326     c.....save the state of the m/km radio button
0327
0328            write(CharString,*) rDistanceSelection
0329            ItemText = CharString
0330            RezHndl.bhdl = GetResource( %val('STR '), %val(int2(rOldDistRadio)))
0331            STRHndl = RezHndl
0332            STRHndl.shdl^.sptr^ = ItemText
0333            call ChangedResource( %val(RezHndl) )
0334            call WriteResource  ( %val(RezHndl) )
0335
0336     c.....save the state of the sec/min/hr radio button
0337
0338            write(CharString,*) rTimeSelection
0339            ItemText = CharString
0340            RezHndl.bhdl = GetResource( %val('STR '), %val(int2(rOldTimeRadio)))
0341            STRHndl = RezHndl
0342            STRHndl.shdl^.sptr^ = ItemText
0343            call ChangedResource( %val(RezHndl) )
0344            call WriteResource  ( %val(RezHndl) )
0345
0346     c.....save the data fork
0347
0348            do i = 1, ntrpts
0349              write(10,*) TofTab(i), LngTab(i), LatTab(i), AltTab(i), JmpTab(i)
0350            end do
0351
0352     c.....close the resource file
0353
0354            call CloseResFile( %val(RefNum) )
0355
0356     c.....close the data fork
0357
0358            close(10)
0359
0360     c.....save the Plot2D-format data file for external use
0361
0362            open( unit=12, file='BDPS Plot Data', status='new', form='unformatted',
0363          &       creator='MDoF', filetype='BINA', recordtype='stream' )
0364
0365     c               123456789-123456
0366            Name(1) = 'Time             '
0367            Name(2) = 'Latitude         '
0368            Name(3) = 'Longitude        '
0369            Name(4) = 'Altitude         '
0370            Name(5) = 'Ground Range     '
0371            Name(6) = 'Wind Azimuth     '
0372            Name(7) = 'Wind Velocity    '
0373
0374            write(12) int4(7), int4(-1), int4(1)
0375            write(12) ( Name(i), i=1,7 )
0376
0377            do i=1,No_of_Pts
0378              write(12)      Time_Array( i ),
0379          &                  LAT_ARRAY ( i ),
0380          &                  LON_ARRAY ( i ),
0381          &                  ALT_ARRAY ( i ),
0382          &                  GRANGE_ARRAY ( i ),
0383          &                  WINDAZ_ARRAY ( i ),
0384          &                  WIND_VEL_ARRAY ( i )
```

```
0385        enddo
0386
0387        close(12)
0388
0389   c.....reset cursor to arrow
0390
0391        call SetCursor ( %val(QDG^.Arrow) )
0392
0393        return
0394        end



0001   c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003   !!G toolbox2.finc
0004
0005   c.....Load the ToolBox traps
0006
0007   !!M Inlines.f
0008
0009   !!S SaveTheMap
0010   c-------------------------------------------------------------------------------
0011        subroutine SaveTheMap
0012   c-------------------------------------------------------------------------------
0013   c      save the contents of the graphics window into a Pict file
0014
0015   c.....pointer for QuickDraw globals
0016
0017        common  / QDGPtr /        QDG
0018        pointer / QDGlobals /     QDG
0019
0020   c.....cursor handle
0021
0022        record / CursHandle /     CursorHndl
0023
0024   c.....Picture record handle and pointer
0025
0026        common / pict   /         PictHndl
0027        record / PicHandle /      PictHndl
0028        record / PicPtr /         PictPtr
0029
0030   c.....reply record
0031
0032        record /SFReply/          SFR
0033
0034   c.....prompt location
0035
0036        record /Point/            PrmptPt
0037
0038   c.....various character strings
0039
0040        string*255               Prompt
0041        string*255               pfName
0042
0043   c.....Define a FORTRAN style parameter corresponding to the Pascal
0044   c      defined constant nil
0045
0046        integer*4                nil
0047
0048   c.....declare integer*2 for Operating System error
0049
0050        integer*2                iOSErr
0051
0052   c.....declare integer array used as header for Pict file
```

```
0053
0054          integer*2                        PictHeader(256)
0055
0056    c.....do loop indices
0057
0058          integer*4                  i4min
0059          integer*4                  i4max
0060          integer*4                  i4
0061
0062    c.....declare temporary integer storage
0063
0064          integer*4                  ndex4
0065          integer*4                  noff4
0066          integer*4                  itemp4
0067          integer*2                  itemp2
0068          integer*1                  itemp1
0069
0070    c.....set the Pict file header array
0071
0072          data PictHeader /
0073        .  $5049, $4354, $0000, $0000, $0000, $0000, $0000, $0000,
0074        1  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0075        2  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0076        3  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0077        4  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0078        5  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0079        6  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0080        7  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0081        8  $0000, $0000, $0000, $0000, $0048, $0000, $0048, $0000,
0082        9  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0083        .  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0084        1  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0085        2  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0086        3  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0087        4  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0088        5  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0089        6  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0090        7  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0091        8  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0092        9  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0093        .  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0094        1  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0095        2  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0096        3  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0097        4  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0098        5  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0099        6  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0100        7  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0101        8  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0102        9  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0103        .  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000,
0104        1  $0000, $0000, $0000, $0000, $0000, $0000, $0000, $0000/
0105
0106    c.....determine name of Pict file via SFPutFile
0107
0108          PrmptPt.H = 62
0109          PrmptPt.V = 64
0110          Prompt    = 'Save current Map as Pict file:'
0111          pfName    = 'Map.pict'
0112          call SFPutFile ( %val(PrmptPt) , %val(Prompt) , %val(pfName) ,
0113        .                  %val(nil) , %ref(SFR) )
0114          pfName = SFR.fName
0115
0116    c.....open Pict file
```

```
0117
0118            if ( SFR.good ) then
0119                iOSErr = SetVol ( %val(nil) , %val(SFR.vRefNum) )
0120                open ( unit=10 , file=pfName , creator='MDPL' , filetype='PICT' ,
0121            .           status='new' , access='sequential' , recordType='stream' ,
0122            .           form= unformatted' )
0123
0124    c........use watch cursor while saving data
0125
0126                cursorHndl = GetCursor ( %val(4) )
0127                call SetCursor ( %val(cursorHndl.CRHDL^.CRPTR^) )
0128
0129    c........pad the Pict file with a header of 256 words
0130
0131                write(10) PictHeader
0132
0133    c........get pointer to Picture record
0134
0135                itemp4  = PicHndl
0136                Pictptr = long ( itemp4 )
0137
0138    c........write number of bytes contained in Picture record
0139
0140                itemp4  = PicPtr
0141                itemp2  = word ( itemp4 )
0142                write(10) itemp2
0143
0144    c........initialize Picture byte counter
0145
0146                ndex4 = itemp2
0147                if ( ndex4.lt.2 ) then
0148                    ndex4 = ndex4 + 65536
0149                end if
0150                ndex4 = ndex4 - 1
0151
0152    c........write bytes of Picture to Pict file byte by byte
0153
0154                itemp4 = PicPtr
0155                do i4 = 2 , ndex4
0156                    itemp1 = byte ( i4 + itemp4 )
0157                    write(10) itemp1
0158                end do
0159
0160    c........continue here if Picture contains too many bytes
0161
0162                noff4 = 0
0163                do while ( itemp1.ne.-1 )
0164                    i4min  = ndex4 + noff4 + 1
0165                    i4max  = ndex4 + noff4 + 65536
0166                    do i4 = i4min , i4max
0167                        itemp1 = byte ( i4 + itemp4 )
0168                        write(10) itemp1
0169                    end do
0170                    noff4 = noff4 + 1
0171                end do
0172
0173    c........set cursor back to arrow before returning
0174
0175                call SetCursor ( %val(QDG^.Arrow) )
0176
0177    c........close Pict file
0178
0179                close ( unit 10 )
0180
```

294

```
0181          end if
0182
0183    c.....kill the pictu
0184
0185          call KillPictu    ( *val(PictHndl) )
0186
0187          return
0188          end



0001    c.....Load a file of   RUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolB   traps
0006
0007    !!M Inlines.f
0008
0009    c------------------   --------------------------------------------------------
0010          subroutine Set  apMenu
0011    c------------------   --------------------------------------------------------
0012
0013    c.....Options menu f   e
0014
0015          include 'MapMe   inc'
0016
0017    c.....local string    ables
0018
0019          string*255           itemName(nMapItems)
0020          string*255           MenuName
0021
0022    c.....set local stri   variables
0023
0024          ItemName(itemG   ewDataSet)   = 'Get new data set/G'
0025          ItemName(itemR   zeTheMap)    = 'Resize the map'
0026          ItemName(itemN   ap)          = 'New map/N'
0027          ItemName(itemSa  Map)         = 'Save map/S'
0028          ItemName(itemRe   aw)         = 'Redraw map/R'
0029          ItemName(itemDo  e)           = 'Quit MapIt/Q'
0030          MenuName                       = 'Map'
0031
0032    c.....get handle for   ap' options menu
0033
0034          if ( iGotMapMen Hndl.eq.0 ) then
0035             MapMenuHndl     = NewMenu ( *val(MapMenuID) , *val(MenuName) )
0036             iGotMapMenuH il = 1
0037          end if
0038
0039    c.....append the men   tems
0040
0041          do i = 1 , nMa  ems
0042             call Append  u ( *val(MapMenuHndl) , *val(ItemName(i)) )
0043          end do
0044
0045    c.....disable the me   tems not initially available
0046
0047          call DisableI      *val(MapMenuHndl) , *val(itemSaveMap)          )
0048          call DisableI      *val(MapMenuHndl) , *val(itemRedraw)           )
0049
0050    c.....draw blank men   er
0051
0052          call ClearMenu
0053          call DrawMenuB
0054
```

```
0055    c.....clear event buffer
0056
0057         call FlushEvents ( %val(everyEvent) , %val(0) )
0058
0059         return
0060         end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the following code in the Main segment
0010
0011    !!S Main
0012    c--------------------------------------------------------------------------
Segment Main
0013         subroutine SetupTheItem ( theDialog, ItemID, SizeIt, ShowIt, EnableIt,
0014         .                          SetTheMax, thePosition, ExtraData, StringID )
0015    c--------------------------------------------------------------------------
0016
0017    !!SETC USINGINCLUDES = FALSE
0018         implicit none
0019
0020    c.....dialog pointer
0021
0022         record / DialogPtr / theDialog
0023
0024    c.....item ID and string ID
0025
0026         integer*2 ItemID
0027         integer*2 StringID
0028
0029    c.....Boolean arguments
0030
0031         logical*1 SizeIt
0032         logical*1 ShowIt
0033         logical*1 EnableIt
0034         logical*1 SetTheMax
0035
0036    c.....item location and size
0037
0038         record / Rect / thePosition
0039
0040    c.....extra data
0041
0042         integer*4 ExtraData
0043
0044    c.....working values
0045
0046         record / Rect          tempRect
0047         record / Handle        DItem
0048         record / ControlHandle  CItem
0049
0050         integer*2  DType
0051
0052         string*255 sTemp
0053
0054    c--------------------------------------------------------------------------
0055
```

296

```
0056    c.....get the item handle and size
0057
0058         call GetDItem( %val(theDialog), %val(ItemID),
0059    &                   %ref(DType), %ref(DItem), %ref(tempRect) )
0060         CItem.CtlH = DItem.bhdl
0061
0062    c.....check to resize all CDEF connected controls
0063
0064         if (SizeIt) then
0065            call SizeControl( %val(CItem), %val(tempRect.right  - tempRect.left),
0066    &                                      %val(tempRect.bottom - tempRect.top) )
0067         endif
0068
0069    c.....pass back the location and size
0070
0071         thePosition = tempRect
0072
0073         if (ExtraData.n .0) then
0074    c.......ignore ExtraData for now
0075            continue
0076         endif
0077
0078    c.....see if CDEF needs the title set again
0079
0080         if(StringID.ne.   then
0081            call GetIndString( %val(sTemp), %val(StringID), %val(1) )
0082            call SetCTitl ( %val(CItem), %val(sTemp) )
0083         endif
0084
0085    c.....see if enable   diable the item
0086
0087         if(EnableIt) th n
0088            call HiliteC trol( %val(CItem), %val(0) )
0089         else
0090            call HiliteC trol( %val(CItem), %val(255) )
0091         endif
0092
0093    c.....see if set the max
0094
0095         if(SetTheMax) then
0096            call SetCtlMax( %val(CItem), %val(12345) )
0097         endif
0098
0099    c.....see if show it to activate it
0100
0101         if(ShowIt) then
0102            call ShowControl( %val(CItem) )
0103         endif
0104
0105         return
0106         end


0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c----------------------------------------------------------------------
0010         subroutine SetUpTheMap
0011    c----------------------------------------------------------------------
```

```
0012    c      Interface with the user to obtain map characteristics.
0013
0014    c.....include common block definition files
0015
0016           include 'CrvDat.inc'
0017           include 'DefLim.inc'
0018           include 'FntCom.inc'
0019           include 'MapCom.inc'
0020           include 'MapLim.inc'
0021           include 'TicDat.inc'
0022           include 'TrjLim.inc'
0023
0024    c.....item stuff
0025
0026           record / handle /       ItHndl
0027           record / rect /         ItRect
0028           integer*4               ItType
0029           integer*2               ItNmbr
0030           string*255              ItText
0031
0032    c....."get Map data" dialog interface records
0033
0034           common / MapSetUp /      MapSetUpPtr,      iGotMapSetUp
0035           record / DialogPtr /     MapSetUpPtr
0036           integer*2               iGotMapSetUp
0037
0038    c.....character strings
0039
0040           character*255           ChrDat
0041
0042    c.....dialog interface variables ( note that pointers are i*4 )
0043
0044           integer*4               infront
0045
0046    c.....dialog interface values
0047
0048           data    infront / -1 /
0049
0050    c.....set dialog font to Times ( it is the most compact )
0051
0052           FntNam = 'Times'
0053           call GetFNum ( val(FntNam) , FntNum )
0054           call setDAfont ( %val(FntNum) )
0055
0056    c.....Get map set up dialog
0057
0058           if ( iGotMapSetUp.eq.0 ) then
0059              MapSetUpPtr  = GetNewDialog ( %val(134) , %val(nil) , %val(inFront) )
0060              iGotMapSetUp = 1
0061           end if
0062           call SetPort ( %val(MapSetUpPtr) )
0063
0064    c.....bring the dialog window to the front
0065
0066           call ShowWindow ( %val(MapSetUpPtr) )
0067           call SelectWin  ( %val(MapSetUpPtr) )
0068
0069    c.....Highlight the QUIT button
0070
0071           ItNmbr = 1
0072           call GetDItem ( val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0073          .                %ref(ItHndl) , %ref(ItRect) )
0074           call PenSize ( val(3) , %val(3) )
0075           call InsetRect ( %ref(ItRect) , %val(-4) , %val(-4) )
```

298

```
0076          call FrameRoundRect ( %ref(ItRect) , %val(18) , %val(18) )
0077
0078   c.....frame the RESET MAP LIMITS button along with the FULL GLOBAL MAP and
0079   c       SCALE TO DATA radio controls
0080
0081          call PenSize ( %val(1) , %val(1) )
0082          call GetDItem ( %val(MapSetUpPtr) , %val(21) , %ref(ItType) ,
0083          .               %ref(ItHndl) , %ref(ItRect) )
0084          myLeft = ItRect.left   - 11
0085          myRite = ItRect.right  + 10
0086          myTop  = ( ItRect.top + ItRect.bottom )/2
0087          call GetDItem ( %val(MapSetUpPtr) , %val(23) , %ref(ItType) ,
0088          .               %ref(ItHndl) , %ref(ItRect) )
0089          myBot  = ItRect.bottom + 10
0090          call MoveTo ( %val(myLeft+8) , %val(myTop) )
0091          call LineTo ( %val(myLeft  ) , %val(myTop) )
0092          call LineTo ( %val(myLeft  ) , %val(myBot) )
0093          call LineTo ( %val(myRite  ) , %val(myBot) )
0094          call LineTo ( %val(myRite  ) , %val(myTop) )
0095          call LineTo ( %val(myRite-8) , %val(myTop) )
0096
0097   c.....display the text items containing map limits
0098
0099          call DisplayMapLimits
0100
0101   c.....set the 'draw grid lines' control
0102
0103          ItNmbr = 3
0104          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0105          .               %ref(ItHndl) , %ref(ItRect) )
0106          call SetCtlValue ( %val(ItHndl) , %val(GridLines) )
0107
0108   c.....set the 'draw time tics' control
0109
0110          ItNmbr = 4
0111          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0112          .               %ref(ItHndl) , %ref(ItRect) )
0113          call SetCtlValue ( %val(ItHndl) , %val(TimeTics) )
0114
0115   c.....set the 'full global map' radio control
0116
0117          ItNmbr = 22
0118          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0119          .               %ref(ItHndl) , %ref(ItRect) )
0120          call SetCtlValue ( %val(ItHndl) , %val(1-LimitType) )
0121
0122   c.....set the 'scale to data' radio control
0123
0124          ItNmbr = 23
0125          call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0126          .               %ref(ItHndl) , %ref(ItRect) )
0127          call SetCtlValue ( %val(ItHndl) , %val(LimitType) )
0128
0129   c.....loop until either the PLOT button or the RETURN button is clicked.
0130   c       Monitor all other relevant events and update the dialog as necessary.
0131
0132          ItNmbr = 0
0133          do while ( ItNmbr.ne.1 .and. ItNmbr.ne.2 )
0134
0135   c.........get number of item hit
0136
0137             call ModalDialog ( %val(nil) , ItNmbr )
0138
0139   c.........alter the 'draw grid lines' user item
```

```
0140
0141            if ( ItNmbr.eq.3 ) then
0142              call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0143        .                     %ref(ItHndl) , %ref(ItRect) )
0144            GridLines = 1 - GetCtlValue ( %val(ItHndl) )
0145            call SetCtlValue ( %val(ItHndl) , %val(GridLines) )
0146          end if
0147
0148  c........alter the 'draw time tics' user item
0149
0150            if ( ItNmbr.eq.4 ) then
0151              call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0152        .                     %ref(ItHndl) , %ref(ItRect) )
0153            TimeTics = 1 - GetCtlValue ( %val(ItHndl) )
0154            call SetCtlValue ( %val(ItHndl) , %val(TimeTics) )
0155            if ( TimeTics.eq.1 ) then
0156              write(ChrDat,*) tDivMj
0157            else
0158              ChrDat = ' '
0159            end if
0160            ItNmbr = 13
0161            ItText = ChrDat
0162            call GetDItem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0163        .                   %ref(ItHndl) , %ref(ItRect) )
0164            call SetIText ( %val(ItHndl) , %val(ItText) )
0165          end if
0166
0167  c........toggle the 'full global map' and 'scale to data' radio controls
0168
0169            if ( ItNmbr.eq.22 .or. ItNmbr.eq.23 ) then
0170            LimitType = 1 - LimitType
0171            call GetDItem ( %val(MapSetUpPtr) , %val(22) , %ref(ItType) ,
0172        .                   %ref(ItHndl) , %ref(ItRect) )
0173            call SetCtlValue ( %val(ItHndl) , %val(1-LimitType) )
0174            call GetDItem ( %val(MapSetUpPtr) , %val(23) , %ref(ItType) ,
0175        .                   %ref(ItHndl) , %ref(ItRect) )
0176            call SetCtlValue ( %val(ItHndl) , %val(LimitType) )
0177          end if
0178
0179  c........reset the map limits if the 'reset map limits' button is selected
0180
0181            if ( ItNmbr.eq.21 ) then
0182
0183  c...........use global map limits
0184
0185              if ( LimitType.eq.0 ) then
0186                xMapMn = LngMin
0187                xMapMx = LngMax
0188                xDivMj = LngDivMj
0189                xDivMi = LngDivMi
0190                yMapMn = LatMin
0191                yMapMx = LatMax
0192                yDivMj = LatDivMj
0193                yDivMi = LatDivMi
0194
0195  c...............scale to fit trajectory data
0196
0197              else
0198                call AutoScale ( MinLng , MaxLng , ndivmj , xMapMn , xMapMx ,
0199        .                        xDivMj , xDivMi )
0200                call AutoScale ( MinLat , MaxLat , ndivmj , yMapMn , yMapMx ,
0201        .                        yDivMj , yDivMi )
0202              end if
0203              call DisplayMapLimits
```

```
0204            end if
0205
0206        end do
0207
0208    c.....return to main program if RETURN was clicked
0209
0210        if ( ItNmbr.eq.2 ) then
0211            call DisposDialog ( %val(MapSetUpPtr) )
0212            call exit
0213        end if
0214
0215    c.....get minimum latitude value
0216
0217        ItNmbr = 5
0218        call GetDitem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0219       .                   %ref(ItHndl) , %ref(ItRect) )
0220        call GetIText ( %val(ItHndl) , %val(ItText) )
0221        ChrDat = ItText
0222        if ( ChrDat.ne.' ' ) then
0223            read(ChrDat.*,iostat=ioflag) tmp1
0224            if ( ioflag.ne.0 ) then
0225                tmp1    = 0.0
0226            end if
0227        else
0228            tmp1    = 0.0
0229        end if
0230        yMapMn = tmp1
0231
0232    c.....get maximum latitude value
0233
0234        ItNmbr = 6
0235        call GetDitem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0236       .                   %ref(ItHndl) , %ref(ItRect) )
0237        call GetIText ( %val(ItHndl) , %val(ItText) )
0238        ChrDat = ItText
0239        if ( ChrDat.ne.' ' ) then
0240            read(ChrDat.*,iostat=ioflag) tmp1
0241            if ( ioflag.ne.0 ) then
0242                tmp1    = 0.0
0243            end if
0244        else
0245            tmp1    = 0.0
0246        end if
0247        yMapMx = tmp1
0248
0249    c.....get latitude major division size
0250
0251        ItNmbr = 7
0252        call GetDitem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0253       .                   %ref(ItHndl) , %ref(ItRect) )
0254        call GetIText ( %val(ItHndl) , %val(ItText) )
0255        ChrDat = ItText
0256        if ( ChrDat.ne.' ' ) then
0257            read(ChrDat.*,iostat=ioflag) tmp1
0258            if ( ioflag.ne.0 ) then
0259                tmp1    = 0.0
0260            end if
0261        else
0262            tmp1    = 0.0
0263        end if
0264        yDivMj = tmp1
0265
0266    c.....get latitude minor division size
0267
```

```
0268            ItNmbr = 8
0269            call GetDitem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0270          .                 %ref(ItHndl) , %ref(ItRect) )
0271            call GetIText ( %val(ItHndl) , %val(ItText) )
0272            ChrDat = ItText
0273            if ( ChrDat.ne.' ' ) then
0274               read(ChrDat,*,iostat=ioflag) tmpl
0275               if ( ioflag.ne.0 ) then
0276                  tmpl  = 0.0
0277               end if
0278            else
0279               tmpl  = 0.0
0280            end if
0281            yDivMi = tmpl
0282
0283    c.....get minimum longitude value
0284
0285            ItNmbr = 9
0286            call GetDitem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0287          .                 %ref(ItHndl) , %ref(ItRect) )
0288            call GetIText ( %val(ItHndl) , %val(ItText) )
0289            ChrDat = ItText
0290            if ( ChrDat.ne.' ' ) then
0291               read(ChrDat,*,iostat=ioflag) tmpl
0292               if ( ioflag.ne.0 ) then
0293                  tmpl  = 0.0
0294             ' end if
0295            else
0296               tmpl  = 0.0
0297            end if
0298            xMapMn = tmpl
0299
0300    c.....get maximum longitude value
0301
0302            ItNmbr = 10
0303            call GetDitem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0304          .                 %ref(ItHndl) , %ref(ItRect) )
0305            call GetIText  %val(ItHndl) , %val(ItText) )
0306            ChrDat = ItText
0307            if ( ChrDat.ne.' ' ) then
0308               read(ChrDat,*,iostat=ioflag) tmpl
0309               if ( ioflag.ne.0 ) then
0310                  tmpl  = 0.0
0311               end if
0312            else
0313               tmpl  = 0.0
0314            end if
0315            xMapMx = tmpl
0316
0317    c.....get longitude major division size
0318
0319            ItNmbr = 11
0320            call GetDitem  %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0321          .                 %ref(ItHndl) , %ref(ItRect) )
0322            call GetIText   %val(ItHndl) , %val(ItText) '
0323            ChrDat = ItTe.
0324            if ( ChrDat.n    '   then
0325               read(ChrDat   iostat=ioflag) tmpl
0326               if ( ioflag ne.0  then
0327                  tmpl    =
0328               end if
0329            else
0330               tmpl    = 0
0331            end if
```

```
0332          xDivMj = tmp1
0333
0334     c......get longitude minor division size
0335
0336          ItNmbr = 12
0337          call GetDitem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0338          .                %ref(ItHndl) , %ref(ItRect) )
0339          call GetIText ( %val(ItHndl) , %val(ItText) )
0340          ChrDat = ItText
0341          if ( ChrDat.ne. ' ) then
0342             read(ChrDat.*,iostat=ioflag) tmp1
0343             if ( ioflag.ne.0 ) then
0344                tmp1   = 0.0
0345             end if
0346          else
0347             tmp1   = 0.0
0348          end if
0349          xDivMi = tmp1
0350
0351     c......get time tic increment
0352
0353          ItNmbr = 13
0354          call GetDitem ( %val(MapSetUpPtr) , %val(ItNmbr) , %ref(ItType) ,
0355          .                %ref(ItHndl) , %ref(ItRect) )
0356          call GetIText ( %val(ItHndl) , %val(ItText) )
0357          ChrDat = ItText
0358          if ( ChrDat.ne. ' ) then
0359             read(ChrDat.*,iostat=ioflag) tmp1
0360             if ( ioflag.ne.0 ) then
0361                tmp1   = 0.0
0362             end if
0363          else
0364             tmp1   = 0.0
0365          end if
0366          tDivMj = tmp1
0367
0368     c......hide dialog
0369
0370          call HideWindow ( %val(MapSetUpPtr) )
0371
0372          return
0373          end


0001     c......Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003     ''G toolbox2.finc
0004
0005     c......Load the Toolb: traps
0006
0007     ''M Inlines.f
0008
0009     c......Put the follow  g code in the Main segment
0010
0011     ''S Main
0012     c------------------- ---------------------------------------------
Segment Main
0013          subroutine SF  AscentFile ( iopen , FilNam )
0014     c----------------- --------------------------------- -----------
0015     c    Select an asc  profile using the unit number [icunit].  A successful
0016     c    open will be : nalled by returning [iopen=1].
0017
0018     c......prompt string
0019
```

```
0020        string*255            Prompt
0021
0022    c.....reply record
0023
0024        record /SFReply/      SFR
0025
0026    c.....prompt location
0027
0028        record /Point/        PrmptPt
0029
0030    c.....file information parameters
0031
0032        record /SFTypeList/   SFT
0033        record /FInfo/        fndrInfo
0034        character*4           FilTyp
0035        character*4           fMaker
0036        string*255            FilNam
0037
0038    c.....I/O error flags
0039
0040        integer*2             ioserr,       iopen
0041
0042    c----------------------------------------------------------------------------
0043
0044    c.....initialize open status flag to zero
0045
0046        iopen  = 0
0047
0048    c.....set the prompt
0049
0050        prompt = 'Select an ascent profile'
0051
0052    c.....set prompt box location
0053
0054        PrmptPt.H = 82
0055        PrmptPt.V = 64
0056
0057    c.....display files of type ascp ("Ascent Profile") or text
0058
0059        SFT.SFT(0).OST = 'ascp'
0060        SFT.SFT(1).OST = 'TEXT'
0061
0062    c.....call the Toolbox
0063
0064        call SFGetFile( %val(PrmptPt), %ref(Prompt), %val(nil),
0065       &                %val(int2(2)), %ref(SFT), %val(nil), %ref(SFR) )
0066
0067    c.....Open the file if user selected open
0068
0069        if ( SFR.good    then
0070
0071          iopen  = 1
0072          FilNam = SFR.fName
0073          FilTyp = SFR.fType.OST
0074          ioserr = GetFInfo ( %val(FilNam) , %val(SFR.vRefNum) , %ref(fndrInfo) )
0075          fMaker = fndrInfo.fdCreator.OST
0076
0077        else
0078
0079          call SysBeep ( %val(int2(20)) )
0080          call ExitToShell
0081
0082        end if
0083
```

304

```
0084          return
0085          end


0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c------------------------------------------------------------------------
0010          subroutine SFOpenFile ( iounit , iopen , FilTyp , fMaker )
0011    c------------------------------------------------------------------------
0012    c     Open an ASCII file using the unit number [iounit].  A successful open will
0013    c     be signalled by returning [iopen=1].
0014
0015    c.....prompt string
0016
0017          string*255          Prompt
0018
0019    c.....reply record
0020
0021          record /SFReply/    SFR
0022
0023    c.....prompt location
0024
0025          record /Point/      PrmptPt
0026
0027    c.....file information parameters
0028
0029          record /SFTypeList/   SFT
0030          record /FInfo/        fndrInfo
0031          character*(*)         FilTyp
0032          character*(*)         fMaker
0033          string*255            FilNam
0034
0035    c.....I/O error flags
0036
0037          integer*2           ioserr,       ios
0038
0039    c------------------------------------------------------------------------
0040
0041    c.....initialize open status flag to zero
0042
0043          iopen  = 0
0044
0045    c.....set the prompt
0046
0047          prompt = 'Select binary data file'
0048
0049    c.....set prompt box location
0050
0051          PrmptPt.H = 8:
0052          PrmptPt.V = 6:
0053
0054    c.....display files of type TEXT or r*df
0055
0056          SFT.SFT(0).OST = 'TEXT'
0057          SFT.SFT(1).OST = 'rtdf'
0058
0059    c.....call the ToolB..
0060
```

```
0061          call SFGetFile  %val(PrmptPt), %ref(Prompt), %val(nil),
0062       &                  %val(2), %ref(SFT), %val(nil), %ref(SFR) )
0063
0064    c.....Open the file   user selected open
0065
0066          if ( SFR.good   then
0067             ioserr = Se  ol ( %val(nil) , %val(SFR.vRefNum) )
0068             open ( unit   unit , file=SFR.fName , status='old' , iostat=ios )
0069             if ( ios.ne   ) then
0070                write(*,   'Error opening',SFR.fName,': ios=',ios
0071             else
0072                iopen  =
0073                FilNam =   R.fName
0074                FilTyp =   FR.fType.OST
0075                ioserr =   stFInfo ( %val(FilNam) , %val(SFR.vRefNum) , %ref(fndrInfo) )
0076                fMaker =   fndrInfo.fdCreator.OST
0077             end if
0078          end if
0079
0080          return
0081          end



0001    c.....Load a file of STRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the ToolB . traps
0006
0007    !!M Inlines.f
0008
0009    c-------------------  --------------------------------------------------------
0010          subroutine SF  nMissionFile ( iopen , FilTyp , fMaker )
0011    c-------------------  --------------------------------------------------------
0012    c    Open a mission  ile using the unit number [iounit].  A successful open
0013    c    will be signal ed by returning [iopen=1].
0014
0015          include 'FileInfo.inc'
0016
0017    c.....prompt string
0018
0019          string*255           Prompt
0020
0021    c.....reply record
0022
0023          record /SFReply      SFR
0024
0025    c.....prompt location
0026
0027          record /Point        PrmptPt
0028
0029    c.....file informati  parameters
0030
0031          record /SFType  'r '  SFT
0032          record /FInf          fndrInfo
0033          character*4           FilTyp
0034          character*4           fMaker
0035          string*255           FilNam
0036
0037    c.....I O error flag
0038
0039          integer*2            iosErr,      ios
0040
0041    c-------------------  --------------------------------------------------------
```

```
0042
0043    c.....initialize ope  status flag to zero
0044
0045         iopen  = 0
0046
0047    c.....set the prompt
0048
0049         prompt = 'Select mission data file'
0050
0051    c.....set prompt box location
0052
0053         PrmptPt.H = 82
0054         PrmptPt.V = 64
0055
0056    c.....display files of type 'rtdf'
0057
0058         SFT.SFT(0).OST = 'rtdf'
0059
0060    c.....call the Toolbox
0061
0062         call SFGetFile( val(PrmptPt), %ref(Prompt), %val(int4(nil)),
0063        &                val(int2(2)), %ref(SFT), %val(int4(nil)), %ref(SFR) )
0064
0065    c.....Open the file   user selected open
0066
0067         if ( SFR.good   hen
0068
0069    c.......open the res   ce fork
0070
0071            RefNum = Open sFile( %ref(SFR.fname) )
0072
0073            select case   sError)
0074
0075              case(noErr
0076                iopen  =
0077                FilNam =   R.fName
0078                FilTyp =   R.fType.OST
0079                ioserr =  tFInfo ( %val(FilNam) , %val(SFR.vRefNum) , %ref(fndrInfo) )
0080                fMaker =  drInfo.fdCreator.OST
0081
0082              case defaul
0083                call SysE ep( %val(int2(20)) )
0084                call Exit oShell
0085
0086            end select
0087
0088         end if
0089
0090         return
0091         end



0001    !!s Sundry
0002    c---------------------------------------------------------------------------
0003         block data Sundry
0004    c---------------------------------------------------------------------------
0005    c     initialize various parameters via block data
0006
0007    c.....common block files
0008
0009         include 'CrvDat.inc'
0010         include 'DefLim.inc'
0011         include 'MapCom.inc'
0012         include 'MapLim.inc'
```

307

```
0013          include 'OptFlg.inc'
0014          include 'PicGrp.inc'
0015          include 'TicDat.inc'
0016
0017    c.....various Map curve settings
0018
0019          data      ipvari      /     1,      0,      0,      0/
0020          data      ipvard      /     2,      0,      0,      0/
0021          data      ipvarg      /     0,      0,      0,      0/
0022          data      idrlin      /     1,      1,      1,      1/
0023          data      lintyp      /     1,      2,      3,      4/
0024          data      DshMsk      / $FFFF, $FFF0, $FCFC, $E0E0/
0025          data      idrsym      /     0,      0,      0,      0/
0026          data      symtyp      /     1,      2,      3,      4/
0027          data      ipstep      /     0,      0,      0,      0/
0028          data      ighoff      /     0,      0,      0,      0/
0029
0030    c.....various Map control settings
0031
0032          data      GridLines   /     0/
0033          data      TimeTics    /     0/
0034          data      LimitType   /     0/
0035
0036    c.....user option flags
0037
0038          data      oCycle      /     0/
0039          data      oSave       /     1/
0040          data      oRedraw     /     2/
0041          data      oNew        /     3/
0042          data      oQuit       /     4/
0043
0044    c.....tic mark settings
0045
0046          data      ndivmj      /    10/
0047          data      lticmj      /    10/
0048          data      lticmi      /     5/
0049
0050    c.....picture grouping commands
0051
0052          data      picGroupBeg /   140/
0053          data      picGroupEnd /   141/
0054
0055    c.....global Map limits
0056
0057          data      LngMin      / -180.0 /
0058          data      LngMax      /  180.0 /
0059          data      LngDivMj    /   30.0 /
0060          data      LngDivMi    /   10.0 /
0061
0062          data      LatMin      /  -90.0 /
0063          data      LatMax      /   90.0 /
0064          data      LatDivMj    /   30.0 /
0065          data      LatDivMi    /   10.0 /
0066
0067    c.....default Map limits
0068
0069          data      xMapMn      / -180.0
0070          data      xMapMx      /  180.0 /
0071          data      xDivMj      /   30.0 /
0072          data      xDivMi      /   10.0
0073
0074          data      yMapMn      /  -90.0 /
0075          data      yMapMx      /   90.0 /
0076          data      yDivMj      /   30.0 /
```

308

```
0077        data      yDir  ι       /   10.0 /
0078
0079        data      tMa;  ¬       /    0.0 /
0080        data      tMa; ω        / 1000.0 /
0081        data      tDi  ⌐        /  100.0 /
0082
0083        end


0001    c.....Load a file of  IRUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
0004
0005    c.....Load the Tool¦   traps
0006
0007    !!M Inlines.f
0008
0009    c.....Put the follow⌐ g code in the Main segment
0010
0011    !!S Main
0012
0013    c-------------------- ------------------------------------------------------
Segment Main
0014        logical functi¬  TrapAvailable( tNumber, tType)
0015    c------------------- ------------------------------------------------------
0016    c     This is my imp  ¬entation of the TrapAvailable function
0017    c     from the DTS S¬ ¬le application.
0018
0019        implicit none
0020        integer*2 tNum        ! trap number, defined in traps.f
0021        integer*1 tTyp¬      ! trap type (enumerated in c/pascal)
0022        integer*1 Tool  ¬p  ! trap type for comparison
0023        data ToolTrap   ¦ / ! second enumerated type should be "1"?
0024        logical check¦  ¬neck2
0025
0026    c.....common block ¬ ¬nition files
0027
0028        include 'Globa  .inc'
0029
0030    c-------------------- ------------------------------------------------------
0031
0032        if ( (tType.e¬ ¦olTrap) .and.
0033    1     ((gMac.mac¦_neType).gt.envMachUnknown) .and.
0034    2     ((gMac.mac¦_neType).lt.envMacII)) then
0035            tNumbe¬ = iand(tNumber,$03FF)
0036            if(tNu¦ ¬¬.gt.$01FF) tNumber=tUnimplemented
0037        end if
0038
0039        check1 = NGet¦_Address(tNumber, tType)
0040        check2 = Get¦_¦ddress(tUnimplemented)
0041        if(check1.¬q.. ¬k2)then
0042           TrapAvai¦¬ ¬ = .false.
0043        else
0044           TrapAvai¦¬ ¬ = .true.
0045        endif
0046
0047        return
0048        end


0001    c.....Load a file of  ¬RUCTURE and PARAMETER definitions at compile time
0002
0003    !!G toolbox2.finc
```

309

```
0004
0005    c.....Load the ToolBox traps
0006
0007    !!M Inlines.f
0008
0009    c-----------------------------------------------------------------------
0010          subroutine TrashBitMap ( oldOffScreen , myBitH )
0011    c-----------------------------------------------------------------------
0012    c     get rid of everything associated with the off screen bit map
0013
0014          record / GrafPtr /         oldOffScreen
0015          record / Handle /          myBitH
0016
0017    c.....close the bit map port and dispose of the associated pointers
0018
0019          call ClosePort    ( %val(oldOffScreen) )
0020          call DisposPtr    ( %val(oldOffScreen.GrafP^.portBits.baseAddr) )
0021          call DisposPtr    ( %val(oldOffScreen) )
0022          call HUnlock      ( %val(myBitH) )
0023          call DisposHandle ( %val(myBitH) )
0024
0025          return
0026          end
```

10.4 BDPS REZ SOURCE CODE

This section contains the BDPS file for input to the MPW resource compiler
("Rez"). Most of the information in this file is directly understandable upon
inspection. However, for the sake of completeness, the file also contains machine-
readable representations of several icons, a "PICT" file, and a popup-menu
handler, all of which are used in building and operating the BDPS program.

```
/*  bdps.r
        Rez file for balloon drift pattern simulation */

#include "systypes.r"
#include "types.r"

#define rMenuBar        128     /* application's menu bar */
#define mApple          128     /* Apple menu */
#define mFile           129     /* File menu */
#define mEdit           130     /* Edit menu */
#define mMap            131     /* Map menu */


#define rAboutAlert     128     /* "About BDPS" alert box */
#define rRunStatus      129     /* Run status dialog */
#define rMapSetup       134     /* Map setup dialog */
#define rResizeMap      135     /* Resize Map dialog */
#define rRunSetupDLOG   256     /* Run Setup dialog */
#define rBalloonPICT    1001    /* PICT for "About BDPS" */


/* these #defines are used to set enable/disable flags of a menu */

#define AllItems        0b1111111111111111111111111111111   /* 31 flags */
#define NoItems         0b0000000000000000000000000000000
#define MenuItem1       0b0000000000000000000000000000001
#define MenuItem2       0b0000000000000000000000000000010
#define MenuItem3       0b0000000000000000000000000000100
#define MenuItem4       0b0000000000000000000000000001000
#define MenuItem5       0b0000000000000000000000000010000
```

```
#define MenuItem6          0b00000000000000000000000000100000
#define MenuItem7          0b00000000000000000000000001000000
#define MenuItem8          0b00000000000000000000000010000000
#define MenuItem9          0b00000000000000000000000100000000
#define MenuItem10         0b00000000000000000000001000000000
#define MenuItem11         0b00000000000000000000010000000000
#define MenuItem12         0b00000000000000000000100000000000
#define     MenuItem13     0b00000000000000000001000000000000    /* 13 */
#define     MenuItem14     0b00000000000000000010000000000000    /* 14 */
#define     MenuItem15     0b00000000000000000100000000000000    /* 15 */
#define     MenuItem16     0b00000000000000001000000000000000    /* 16 */
#define     MenuItem17     0b00000000000000010000000000000000    /* 17 */
#define     MenuItem18     0b00000000000000100000000000000000    /* 18 */
#define     MenuItem19     0b00000000000001000000000000000000    /* 19 */
#define     MenuItem20     0b00000000000010000000000000000000    /* 20 */
#define     MenuItem21     0b00000000000100000000000000000000    /* 21 */
#define     MenuItem22     0b00000000001000000000000000000000    /* 22 */
#define     MenuItem23     0b00000000010000000000000000000000    /* 23 */
#define     MenuItem24     0b00000000100000000000000000000000    /* 24 */
#define     MenuItem25     0b00000001000000000000000000000000    /* 25 */
#define     MenuItem26     0b00000010000000000000000000000000    /* 26 */
#define     MenuItem27     0b00000100000000000000000000000000    /* 27 */
#define     MenuItem28     0b00001000000000000000000000000000    /* 28 */
#define     MenuItem29     0b00010000000000000000000000000000    /* 29 */
#define     MenuItem30     0b00100000000000000000000000000000    /* 30 */
#define     MenuItem31     0b01000000000000000000000000000000    /* 31 */


type 'MDoF' as 'STR ';                /*  crcH is the signature */
resource 'MDoF' (0)  {                /* the creator resource ID must be zero */
        "Balloon Drift Patter: Simulation 1.0 copyright 1991"
};


/* use an MBAR resource to c. veniently load all menus */

resource 'MBAR' (rMenuBar. ": lloon Drift menu bar", preload) {
        { mApple, mFile, mEdi  mMap};              /* four menus */
};
resource 'MENU' (mApple, "Ap e menu", preload) {
        mApple, textMenuProc,
        AllItems & ~MenuItem1  '* disable dashed line, enable About and DAs */
        enabled, apple,
        {
                "About BDPS..
                        noico: okey, nomark, plain;
                "-",
                        noicon okey, nomark, plain
        }
};

resource 'MENU' (mFile, "Fil menu", preload) {
        mFIle, textMenuProc
        MenuItem12,                                          /* enable Quit only, program enables
others */
        enabled, "File",
        {
                "New Missic
                        noico: "", nomark, plain;
                "Open Missic
                        noicc: "". nomark, plain;
                "-",
                        noico: key, nomark, plain;
                "Close",
                        noico: "". nomark, plain;
                "Save",
```

```
                         noicon    3", nomark, plain;
              "Save As...",
                         noicon    :key, nomark, plain;
              "Revert",
                         noicon    :key, nomark, plain;
              "-",
                         noicon    :key, nomark, plain;
              "Page Setup..."
                         noicon    :key, nomark, plain;
              "Print...",
                         noicon    :key, nomark, plain;
              "-",
                         noicon    :key, nomark, plain;
              "Quit",
                         noicon    2", nomark, plain
         }
};

resource 'MENU' (mEdit, "Ed.  enu", preload) {
         mEdit, textMenuProc,
         NoItems,                  disable everything, program does the enabling */
         enabled, "Edit",
          {
              "Undo",
                         noicon    ". nomark, plain;
              "-",
                         noicon    key, nomark, plain;
              "Cut",
                         noicon    <", nomark, plain;
              "Copy",
                         noicon    :", nomark, plain;
              "Paste",
                         noicon    V", nomark, plain;
              "Clear",
                         noicon    :key, nomark, plain
         }
};

resource 'MENU' (mMap, "Map   nu", preload) {
         mMap, textMenuProc,
         NoItems,                  * disable everything, program does the enabling */
         enabled, "Map",
          {
              "Get New Data   :",
                         noicon    :okey, nomark, plain;
              "Resize the Mc  ,
                         noicon.   .okey, nomark, plain;
              "New Map",
                         noicon    :okey, nomark, plain;
              "Save Map",
                         noicon.   okey, nomark, plain;
              "Redraw",
                         noicon    :key, nomark, plain;
              "Done",
                         noicon    :key, nomark, plain
         }
};

resource 'MENU' (41,"Wind M   .:",preload) {
         41, textMenuProc,

         AllItems,                           /* Enable and disable of items */
         enabled, "Wind Model                /* Menu list name */
```

```
        {

        /*  1 */  "January Climate     ",

        noIcon, noKey, noMark,  plain,

        /*  2 */  "February Climate    ",

        noIcon, noKey, noMark,  plain,

        /*  3 */  "March Climate       ",

        noIcon, noKey, noMark,  plain,

        /*  4 */  "April Climate       ",

        noIcon, noKey, noMark,  plain,

        /*  5 */  "May Climate         ",

        noIcon, noKey, noMark,  plain,

        /*  6 */  "June Climate        ",

        noIcon, noKey, noMark,  plain,

        /*  7 */  "July Climate        ",

        noIcon, noKey, noMark,  plain,

        /*  8 */  "August Climate      ",

        noIcon, noKey, noMark,  plain,

        /*  9 */  "September Climate    ",

        noIcon, noKey, noMark,  plain,

        /* 10 */  "October Climate     ",

        noIcon, noKey, noMark,  plain,

        /* 11 */  "November Climate    ",

        noIcon, noKey, noMark,  plain,

        /* 12 */  "December Climate    ",

        noIcon, noKey, noMark,  plain,

        /* 13 */  "Forecast    ",

        noIcon, noKey, noMark,  plain,

        }

    };

resource 'MENU' (42,"Popup menu",preload) {
        42, textMenuProc,

        AllItems,                               /* Enable and disable of items */
        enabled, "Popup menu".                  /* Menu list name */
```

```
            {
            /*  1 */  "Untitled274    ",

            noIcon, noKey, noMark,  plain,

            }

        };


resource 'MENU' (43,"Input Ascent Profile:",preload) {
        43, textMenuProc,

        AllItems,                                /* Enable and disable of items */
        enabled, "Input Ascent Profile:",   /* Menu list name */
            {

            /*  1 */  "Ascent Profile    ",

            noIcon, noKey, noMark,  plain,

            /*  2 */  "User-defined file...   ",

            noIcon, noKey, noMark,  plain,

            }

        };


resource 'BNDL' (128) {
        'MDoF',               /* the signature of this application */
        0,                    /* the creator resource ID must be 0 */
        {       'ICN#',
                {       0, 128,
                        1, 12
                        2, 13
                        3, 13.
                },
                'FREF',
                {       0, 12
                        1, 12
                        2, 13
                        3, 13
                }
        }
};

resource 'ICN#' (128, "BDPS    plication") {
        {       /* array: 2 e   ents */
                /* [1] */
                $"0000 0000 0    0000 0000 0000 0003 F000"
                $"00BD EC00      3200 0016 3900 0566 3900"
                $"002C 3C80      3C80 02EC 3C80 002C 3C80"
                $"0024 3C80      3000 0016 3900 000A 3A00"
                $"00AF 3400      3800 0002 0800 002E 0800"
                $"0001 F000      1000 0017 1000 0001 F000"
                $"0001 F000      F000 0001 F000 0000 E0",
                /* [2] */
                $"3FFF FFFC    F FFFE FFFF FFFF FFFF FFFF"
                $"FFFF FFFF FF F FFFF FFFF FFFF FFFF FFFF"
                $"FFFF FFFF FF F FFFF FFFF FFFF FFFF FFFF"
                $"FFFF FFFF FF F FFFF FFFF FFFF FFFF FFFF"
```

```
                        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"
                        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"
                        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"
                        $"FFFF FFFF FFFF FFFF 7FFF FFFE 3FFF FFFC"
            ;
    };

resource 'ICN#' (129, "runtime data file") {
        {               /* array: 2 elements */
                        /* [1] */
                        $"FFFF FF00 8010 0180 B81F 8140 A42F 6120"
                        $"B859 9110 A451 C908 B931 CDFC 8161 E4FE"
                        $"B961 E406 A561 E406 A561 E406 A521 E006"
                        $"B8A1 E806 8051 C806 B851 D006 A439 A006"
                        $"B817 4006 A010 4006 A00F 8006 8008 8006"
                        $"9C08 8006 A00F 8006 BC0F 8006 840F 8006"
                        $"B807 0006 8001 0006 A299 A696 B6B3 2AD6"
                        $"AA99 AAB6 A2B2 2C96 FFFF FFFE 7FFF FFFE",
                        /* [2] */
                        $"FFFF FF00 FFFF FF80 FFFF FFC0 FFFF FFE0"
                        $"FFFF FFF0 FFFF FFF8 FFFF FFFC FFFF FFFE"
                        $"FFFF FFFE FFFF FFFE FFFF FFFE FFFF FFFE"
                        $"FFFF FFFE FFFF FFFE FFFF FFFE FFFF FFFE"
                        $"FFFF FFFE FFFF FFFE FFFF FFFE FFFF FFFE"
                        $"FFFF FFFE FFFF FFFE FFFF FFFE FFFF FFFE"
                        $"FFFF FFFE FFFF FFFE FFFF FFFE FFFF FFFE"
                        $"FFFF FFFE FFFF FFFE FFFF FFFE 7FFF FFFE"
        }
    };

resource 'ICN#' (130, "atmosphere data") {
        {               /* array: 2 elements */
                        /* [1] */
                        $"E031 4198 8DCB 1244 E804 0802 1080 000A"
                        $"9080 0011 8880 0811 E2C2 0841 0269 0C21"
                        $"4200 B061 41AA 0AB2 40D5 555C 007A EAA0"
                        $"A00D B6C4 E007 0F00 A000 1048 A004 0180"
                        $"0040 C290 4088 0000 A000 8100 E102 0000"
                        $"A000 0840 0241 0200 E040 2080 4010 0400"
                        $"4404 2000 4020 0200 0100 8000 E008 0000"
                        $"8400 0000 C003 8CE6 8822 5E4F E003 9249",
                        /* [2] */
                        $"0031 C198 0FFB F3FC 1FFF FFFE 1FFF FFFE"
                        $"1FFF FFFF 0FFF FFFF 03FF FFFF 03FF FFFF"
                        $"03FF FFFF 01FF FFFE 00FF FFFC 007F FFE0"
                        $"000F BFC4 0007 0F00 0000 1048 0004 0180"
                        $"0040 C290 0088 0000 0000 8100 0102 0000"
                        $"0000 0840 0241 0200 0040 2000 0010 0000"
                        $"0404 2880 0020 0000 0100 0000 0008 8200"
                        $"0400 0000 0000 0000 0820"
        }
    };

resource 'ICN#' (131, "ascent profile") {
        {               /* array: 2 elements */
                        /* [1] */
                        $"FFFF FFFF 8000 0001 8C66 74B9 9289 4691"
                        $"9EE8 6791 9223 4591 92C6 7491 8000 0001"
                        $"9FFF FFF9 9000 0009 9000 0009 9040 0009"
                        $"90AF FFE9 9100 0009 9100 0009 9300 0009"
                        $"9200 0009 9200 0009 9400 0009 9400 0009"
                        $"9800 0009 9800 0009 9800 0009 9FFF FFF9"
                        $"8000 0001 B9C6 751D A529 4511 B9C9 6519"
                        $"A149 4511 A126 45DD 8000 0001 FFFF FFFF",
```

315

```
                    *  [2]  *
           $"FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFrF"
           $"FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF"
           $"FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFrF"
           $"FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF"
           $"FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF"
           $"FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF"
           $"FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF"
           $"FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF"
        }
};

resource 'FREF' (128) {
        'APPL',
        0,
        ""
};

resource 'FREF' (129) {
        'rtdf',
        1,
        ""
};

resource 'FREF' (130) {
        'gram',
        2,
        ""
};

resource 'FREF' (131) {
        'ascp',
        3,
        ""
};

/* this ALRT and DITL are used as an About screen */

resource 'ALRT' (rAboutAlert, "About Box", purgeable) {
        {40, 20, 338, 506},
        rAboutAlert,
        { /* array: 4 elements */
                /* [1] */
                OK, visible, silent,
                /* [2] */
                OK, visible, silent,
                /* [3] */
                OK, visible, silent,
                /* [4] */
                OK, visible, silent
        }
};

resource 'DITL' (rAboutAlert, "About Box", purgeable) {
        {       /* array DITLarray: 2 elements */
                /* [1] */
                {247, 352, 271, 452},
                Button {
                        enabled
                        "OK"
                },
                /* [2] */
                {-1, -1, 290, 176},
                Picture {
```

316

```
                        disab.
                        rBall      .t
                }
        {
};

 * This is the definition    the run status dialog */

resource 'DLOG' (rRunStatu.  Run Status Dialog") {
        (40, 40, 123, 368).             /* Top Left Bottom Right */
        dBoxProc,
        invisible,
        noGoAway,
        0x0,
        rRunStatus,
        ""
};

/* This is the item list for  he run status dialog */

resource 'DITL' (rRunStatus.  Run Status Dialog") {
        {       /* array DITL ray: 3 elements */
                /* [1] */
                (14, 33, 50.  6),
                StaticText {
                        disabl  ,
                        "Gener  ing balloon drift pattern data..."
                },
                /* [2] */
                (44, 53, 72, 2 3).
                StaticText {
                        disabl  .
                        "Perce   complete:"
                },
                /* [3] */
                (44, 188, 60,  70),
                EditText {
                        enabled.
                        "0.0"
                }
        }
};

/* This is the definition for the run setup dialog,  "Untitled" */

resource 'DLOG' (rRunSetupDLOG, "Run Setup Dialog", purgeable) {  /* Dialog */
        { 30, 10, 332, 501 }            /* Top Left Bottom Right */
        dBoxProc, Invisible, n GoAway,256, rRunSetupDLOG,   /* ProcID, visible, noGoAway,
RefCon. DITL */
        "Untitled"
        };

 * This is the item list for   e run setup dialog*/

resource 'DITL' (rRunSetupDL    "Run Setup Dialog", purgeable)
        (   * DITL array of      tems */

        /* [1] */
        ( 268, 396, 288, 475       /* Top Left Bottom Right */
        Button {
                enabled,
                "Close"
        },
```

```
  *  [2]  *
( 184, 396, 204, 476        '* Top Left Bottom Right */
Button (
        enabled,
        "Run"
,.

  *  [3]  *
( 240, 396, 260, 476        /* Top Left Bottom Right */
Button (
        enabled,
        "Map"
},

  * [4] */
( 212, 396, 232, 476        /* Top Left Bottom Right */
Button (
        enabled,
        "Save"
},

/* [5] */
( 89, 246, 104, 326 ),      /* Top Left Bottom Right */
RadioButton (
        enabled,
        "deg West"
},

/* [6] */
( 89, 338, 104, 418 ),      /* Top Left Bottom Right */
RadioButton (
        enabled,
        "deg East"
},

/* [7] */
( 117, 246, 132, 282 ),     /* Top Left Bottom Right */
RadioButton (
        enabled,
        "m"
},

/* [8] */
( 117, 294, 132, 338 ),     /* Top Left Bottom Right */
RadioButton (
        enabled,
        "km"
},

/* [9] */
( 145, 246, 160, 294 ),     /* Top Left Bottom Right */
RadioButton (
        enabled,
        "sec"
},

/* [10] */
( 145, 294, 160, 342        /* Top Left Bottom Right */
RadioButton (
        enabled,
        "min"
},

/* [11] */
```

318

```
{ 145, 342, 160, 378 },       /* Top Left Bottom Right */
RadioButton {
        enabled,
        "hr"
},

 * [12] *
{ 65, 104, 80, 168 },         /* Top Left Bottom Right */
StaticText {
        disabled,
        "Latitude:"
},

/* [13] */
{ 89, 94, 104, 174 },         /* Top Left Bottom Right */
StaticText {
        disabled,
        "Longitude:"
},

/* [14] */
{ 65, 263, 80, 343 },         /* Top Left Bottom Right */
StaticText {
        disabled,
        "deg"
},

/* [15] */
{ 141, 61, 157, 169 },        /* Top Left Bottom Right */
StaticText {
        disabled,
        "Flight Duration:"
},

/* [16] */
{ 69, 18, 101, 78 },          /* Top Left Bottom Right */
StaticText {
        disabled,
        "Launch Position"
},

/* [17] */
{ 117, 62, 133, 170 },        /* Top Left Bottom Right */
StaticText {
        disabled,
        "Initial Altit  e:"
},

/* [18] */
{ 19, 16, 50, 68 },           /* Top Left Bottom Right */
StaticText {
        disabled,
        "Mission Label
},

/* [19] */
{ 19, 83, 53, 465 }.          /* Top Left Bottom Right */
EditText {
        enabled,
        "Wallops Isla   light - Z configuration (this text may be used to identify the
mission represented by this    )"
},

/* [20] */
```

```
        ( 54, 185, 78, 235              /* Top Left Bottom Right */
        EditText {
                enabled,
                "54.45"
        },

        /* [21] */
        ( 90, 185, 104, 235             /* Top Left Bottom Right */
        EditText {
                enabled,
                "154.45"
        },

        /* [22] */
        { 116, 185, 130, 235            /* Top Left Bottom Right */
        EditText {
                enabled,
                "154.45"
        },

        /* [23] */
        ( 143, 185, 157, 235            /* Top Left Bottom Right */
        EditText {
                enabled,
                "154.45"
        },

        /* [24] */
        ( 175, 93, 195, 343             /* Top Left Bottom Right */
        Control {
                enabled,
                133                             /* Popup Resource ID linked to */
        },

        /* [25] */
        ( 204, 40, 224, 345             /* Top Left Bottom Right */
        Control {
                enabled,
                133                             /* Popup Resource ID linked to */
        },

        }
};

/* This is the definition f    he map setup dialog */

resource 'DLOG' (rMapSetup,     p Setup Dialog", purgeable) {   /* Dialog */
        ( 41, 11, 330, 390              /* Top Left Bottom Right */
        documentProc, visibl-   noGoAway,0x0, rMapSetup,   /* ProcID, visible, noGoAway,
RefCon, DITL */
        "Map Set Up"
        };

/* This is the item list fc   e map setup dialog*/

resource 'DITL' (rMapSetup,     p Set Up", purgeable) {
        {         /* array DIT   y: 23 elements */
                /* [1] */
                (11, 11, 35,
                Button {
                        enabl-
                        "PLOT"
                },
                /* [2] */
```

```
{46, 11. "2.
Button {
        enab_
        "PE::P
},
/* [3] */
{11, 101. 3..        ',
CheckBox {
        enab_a
        "draw    . lines"
},
/* [4] */
{36, 101, 55,        ',
CheckBox {
        enab_a
        "draw    . tics"
},
/* [5] */
{116, 111, 1:    :0},
EditText {
        disab_
        ""
},
/* [6] */
{151, 111, 1"    :0},
EditText {
        disab_
        ""
},
/* [7] */
{186, 111, 21.    :0},
EditText {
        disab_
        ""
},
/* [8] */
{221, 111, 24.    10},
EditText {
        disab_
        ""
},
/* [9] */
{116, 221, 14.    :20},
EditText {
        disab_
        ""
},
/* [10] */
{151, 221, 17:    20},
EditText {
        disabla
        ""
},
/* [11] */
{186, 221, 21.    :20},
EditText {
        disable
        ""
},
/* [12] */
{221, 221, 245.   :20},
EditText {
        disablec.
        ""
```

```
},
/* [13] */
{256, 111, 280, 210},
EditText {
        disabled,
        ""
},
/* [14] */
{116, 6, 140, 105},
StaticText {
        disabled,
        "minimum axis value"
},
/* [15] */
{151, 6, 175, 105},
StaticText {
        disabled,
        "maximum axis value"
},
/* [16] */
{186, 6, 210, 105},
StaticText {
        disabled,
        "major division size"
},
/* [17] */
{221, 6, 245, 105},
StaticText {
        disabled,
        "minor division size"
},
/* [18] */
{261, 6, 285, 105},
StaticText {
        disabled,
        "time tic increment"
},
/* [19] */
{96, 111, 110, 190},
StaticText {
        disabled,
        "latitude"
},
/* [20] */
{96, 221, 110, 300},
StaticText {
        disabled,
        "longitude"
},
/* [21] */
{11, 241, 35, 360},
Button {
        enabled,
        "RESET MAP LIMITS"
},
/* [22] */
{41, 241, 60, 360},
RadioButton {
        enabled,
        "full global map"
},
/* [23] */
{66, 241, 85, 360},
RadioButton {
```

```
                        enabl
                    "scal    data"
                }
            }
};

/* This is the definition f    .e map resize dialog */

resource 'DLOG' (rResizeMap  'esize Map Dialog", purgeable) {   /* Dialog */
        { 41, 11, 170, 370          /* Top Left Bottom Right */
        documentProc, visibl   :GoAway,0x0, rResizeMap,    /* ProcID, visible, noGoAway,
RefCon, DITL */
        "Resize Map"
        };

/* This is the item list fo:   e map resize dialog*/

resource 'DITL' (rResizeMap    esize Map", purgeable) {
        {        /* array DITL    y: 6 elements */
            /* [1] */
            {11, 21, 35.
            Button {
                    enable:
                    "OK"
            },
            /* [2] */
            {46, 21, 70.         .
            Button {
                    enable
                    "RESET
            },
            /* [3] */
            {11, 236, 35,    :},
            EditText {
                    disabl
                    ""
            },
            /* [4] */
            {46, 236, 70,    :},
            EditText {
                    disable
                    ""
            },
            /* [5] */
            {11, 126, 35,    :},
            StaticText {
                    disabl
                    "map w   ow width"
            },
            /* [6] */
            {46, 126, 70.    :},
            StaticText {
                    disabl
                    "map w   w height"
            }
        }
};


resource 'STR#' (281 , "Lat:   _Label") {
{  /*                                       for Static text */
/* [1] */
"Latitude:"
        }
```

```
};

resource 'STR#' (282 , "Lon    e_Label") {
{   /*                                      for Static text */
* [1] *
"Longitude:"
        )
};

resource 'STR#' (284 , "Lat    e_Units") {
{   /*                                      for Static text */
* [1] */
"deg"
        )
};

resource 'STR#' (287 , "Dur    n_Label") {
{   /*                                      for Static text */
/* [1] */
"Flight Duration:"
        )
};

resource 'STR#' (274 , "Laun   Label2") {
{   /*                                      for Static text */
/* [1] */
"Launch Position"
        )
};

resource 'STR#' (290 , "Alti   e_Label") {
{   /*                                      for Static text */
/* [1] */
"Initial Altitude:"
        )
};

resource 'STR#' (305 , "Miss_ n_Label") {
{   /*                                      for Static text */
/* [1] */
"Mission Label:"
        )
};

resource 'STR#' (280 , "Latit  e") {
{   /*                                      for default edit text */
/* [1] */
"54.45"
        )
};

resource 'STR#' (283 , "Latit  e2") {
{   /*                                      for default edit text */
/* [1] */
"154.45"
        )
};

resource 'STR#' (270 , "Latit  e3") {
{   /*                                      for default edit text */
/* [1] */
"154.45"
        )
};
```

```
resource 'STR#' (291 , "Lat.   .e4") {
{   /*                                    for default edit text */
 /* [1] */
"154.45"
       }
};

resource 'STR#' (304 , "Mi;   _Text") {
{   /*                                    for default edit text */
 /* [1] */
"Wallops Island flight - 2    iguration (this text may be used to identify the mission
represented by this data)"
       }
};

resource 'PICT' (rBalloonPI    "Erics About BDPS picture") {
       14210,
       (-1, -1, 286, 508),
       $"1101 A000 82A1 00F   04 000C 0080 0100 0AFF FFFF FF01 1E01 FC09 8822 8822 8822"
       $"8822 3100 FF00 00:   00 3109 FFFF FFFF FFFF FFFF 38A1 00B6 0004 0001 0001 0700"
       $"0000 0023 0000 A1.   00 0400 0C00 8022 0106 0006 0000 A000 A0A1 00A4 0002 070C"
       $"0900 0000 0000 00:   71 0092 0102 0004 011A 002C 0104 0007 0105 0007 0107 0006"
       $"0108 0005 0109 00    0C 0004 010E 0004 0110 0004 0112 0005 0113 0005 0115 0006"
       $"0117 0007 0118 00    1A 0009 011A 0018 011A 0028 0118 0029 0117 002A 0115 002B"
       $"0114 002B 0112 00:   0 002C 010E 002C 010D 002C 010A 002C 0108 002B 0107 002B"
       $"0105 002A 0104 00:   02 0027 0102 0018 0102 0008 0102 0008 0104 0007 0100 0A00"
       $"0000 0000 0000 00    01 0001 09FF FFFF FFFF FFFF FF22 0102 0008 FE04 23FE 0423"
       $"0007 2302 0523 03:   20 0023 1F00 2300 0023 02FC 2302 FB23 00FA 23FE FB23 FDFC"
       $"2300 0023 E100 23    23 0000 0A00 0000 0000 0000 0084 000A 0000 0000 0000 0000"
       $"A000 A301 000A FF:   FF 011E 01FC 8400 0A00 0000 0000 0000 0022 0104 0007 0001"
       $"23FF 0223 FF01 23    23 FF03 2300 0223 0002 2301 0223 0001 2301 0223 0102 2300"
       $"0123 0202 230F 00.   00 2301 FE23 01FF 2301 FE23 00FF 2301 FE23 00FE 2300 FE23"
       $"00FF 2300 FD23 FF:   00 FF23 FFFE 23FF FF23 FEFE 23F1 0023 F000 2300 0023 FF02"
       $"A000 A1A1 00B6 00    01 0001 0700 0000 0023 0000 A100 9600 0C01 0000 0002 0000"
       $"0000 0000 00A1 00:   08 FFFB 0000 0006 0000 0100 0A01 0600 1201 1700 212C 000B"
       $"0010 0850 616C 61    6E 6F03 0010 0D00 1028 0113 0013 0152 A000 97A1 0096 000C"
       $"0100 0000 0200 00.   00 0000 A100 9A00 08FF FB00 0000 0600 0001 000A 0106 0006"
       $"0117 0015 2801 13    01 43A0 0097 A100 9600 0C01 0000 0002 0000 0000 0000 00A1"
       $"009A 0008 FFFB 00    06 0000 0100 0A01 0600 1D01 1700 2C29 1701 43A0 0097 A100"
       $"9600 0C01 0000 00:   00 0000 0000 00A1 009A 0008 0003 0000 003B 0000 0100 0A01"
       $"0200 3301 1D0C AC    0C 2801 0C00 3410 436F 6C65 6D61 6E20 5265 7365 6172 6368"
       $"0D00 0D29 6801 0D'   0C 2801 1A00 3413 4875 6E74 7376 696C 6C65 2044 6976 6973"
       $"696F 6EA0 0097 A1    00 0400 4000 8001 000A FFFF FFFF 011E 01FC 2200 0301 0E00"
       $"00A0 00A0 A100 A4:   01 4001 000A 0000 0000 0000 0000 0700 0100 0122 0009 0116"
       $"F8FA 23E4 FD23 F1    F8 0923 F1FF 23F2 F223 E4FB 23F3 0E23 FF0E 23F8 0423 F901"
       $"23FC 0623 ECFF 23:   23 F2FF 23F9 04A0 00A3 0100 0AFF FFFF FF01 1E01 FC22 0009"
       $"0116 FEFF 23FB FD.   FE 23FA FF23 FD00 23FF 0023 FC00 23FC 0023 FE00 23FD 0123"
       $"FF00 23FC 0223 FF    FF 0123 FE02 23FD 0223 FC03 23FE 0223 FF01 23FF 0123 FE02"
       $"23FD 0123 FD00 23    023 FE00 23FC FF23 FCFE 23FD FE23 FEFE 23FE FE23 FBFD 23FB"
       $"FE23 FBFF 23FC FF    F00 23FB FF23 FD00 23FF 0023 FC01 23FE 0123 FD01 23FF 0123"
       $"FF01 23FF 0123 FDC   23FF 0223 FF04 2300 0223 0001 23FF 0323 FF02 23FE 0323 FF00"
       $"23FE 0123 FC02 23FE   023 FF00 23FF 0023 FF01 23FF 0123 FF01 2300 0023 FD02 23FD"
       $"0123 FD00 23FD 002    500 23FC 0023 FD00 23FD 0123 FF01 23FF 0123 FF00 23FD 0023"
       $"FC00 23FF 0023 FE0    0FC 0123 FD01 23FD 0123 FE01 A000 A1A1 00B6 0004 0001 0001"
       $"0700 0000 0023 000.   00 B600 0400 4000 8022 0011 0104 0000 A000 A0A1 00A4 0002"
       $"0340 0988 2288 223    98 2271 0302 000B 00D5 00B4 0120 000B 0114 000C 0112 000D"
       $"010F 000E 010C 00:   38 0011 0105 0011 0104 0012 0102 0013 0100 0013 0100 0014"
       $"00FE 0017 00FA 00:   0F9 001A 00F6 001E 00F3 0020 00F2 0021 00F2 0022 00F2 0023"
       $"00F2 0024 00F3 002   0F4 0023 00F6 0021 00F9 0020 00FA 0020 00FA 001F 00FB 001F"
       $"00FC 001F 00FD 001F   0FD 001F 00FD 0020 00FD 0022 00FC 0026 00FA 0028 00F8 002B"
       $"00F6 002E 00F4 003   0F2 0038 00F0 003A 00F0 003C 00F0 0041 00EF 0046 00EF 004B"
       $"00EF 004D 00EF 005    0EF 0055 00F0 0059 00F1 005D 00F2 005F 00F3 0061 00F4 0064"
       $"00F6 0067 00F8 006A   0FB 006B 00FC 006D 00FF 0071 0104 0072 0107 0073 010A 0074"
```

```
$"0110 0074 0112 00~   :13 0074 0114 0074 0115 0074 0116 0074 0116 0074 0116 0074"
$"0116 0074 0115 00~   :15 0074 0115 0075 0112 0075 0111 0076 010D 0076 010B 0076"
$"0109 0076 0104 00~   :FF 0074 00FB 0073 00F9 0072 00F6 0072 00F4 0072 00F0 0073"
$"00EF 0074 00EE 00~   ~EE 0078 00ED 0079 00EE 007C 00F0 007D 00F1 007E 00F2 007F"
$"00F5 0080 00F9 00~   :FD 0080 0100 007F 0105 007D 0110 007B 0116 007B 0117 007A"
$"011A 007A 011C 00~   :1E 007A 011F 007A 011F 007C 0120 007E 011F 007E 011F 0080"
$"011E 0083 011B 00~   :17 008B 0112 008D 010F 008E 010E 0090 010B 0092 0109 0094"
$"0107 0096 0106 00~   :05 009A 0104 009C 0104 009D 0104 009F 0104 00A3 0104 00A7"
$"0103 00AB 0102 00A~  :01 00AF 0101 00B0 0100 00B3 00FE 00B3 00FD 00B4 00FC 00B4"
$"00FA 00B4 00F6 00B~  :F5 00B3 00F3 00B2 00F1 00B0 00EC 00AE 00E7 00AD 00E3 00AC"
$"00E0 00AC 00DF 00A~  :DC 00AA 00DB 00AA 00DA 00A9 00D9 00A7 00D8 00A7 00D8 00A6"
$"00D8 00A5 00D8 00A   :D8 00A1 00D8 009D 00D9 009A 00DA 0098 00DA 0094 00DB 008C"
$"00DB 0088 00DB 00B   :DB 0082 00DB 007C 00DA 0077 00DA 0074 00D9 006F 00D8 0064"
$"00D6 005F 00D5 005~  :D5 0051 00D5 004C 00D5 004A 00D5 0045 00D6 0040 00D7 003B"
$"00D8 0038 00D9 003~  :DA 0031 00DC 002B 00DF 0027 00E2 0025 00E3 0022 00E5 001E"
$"00E8 001B 00EC 001   ~EF 0017 00F1 0016 00F3 0013 00F7 0011 00FB 0010 00FF 000F"
$"0102 000E 0106 00C   :0E 000B 0113 0100 0A00 0000 0000 0000 0007 0001 0001 2200"
$"0B01 14F0 0623 F90   :F9 0623 FD0A 2307 FD23 04FC 2302 0223 F311 23FD 1323 0113"
$"2307 0F23 0B09 230   :23 0B00 23FB 0123 EF02 23EE F923 FC09 230A 0823 14FD 2318"
$"F923 F90E 23E8 102   :10 23FA 1223 EBF7 23EE FC23 01F6 2303 F023 00F0 23FC E923"
$"FDEC 2301 EE23 08E~  :0C EF23 10F5 2311 FB23 09FF 0A88 2288 2288 2288 2284 000A"
$"0000 0000 0000 000   :00 A301 000A FFFF FFFF 011E 01FC 8400 0A00 0000 0000 0000"
$"0022 000B 0114 FE0   :FD 0123 FD01 23FC 0223 FD01 23FF 0023 FE01 23FE 0123 0000"
$"23FE 0123 FC03 23F~  :23 FE02 23FD 0423 FF02 2300 0123 0001 2300 0123 0101 2301"
$"FF23 0200 2303 FE2   :FF 2300 0023 01FF 2301 0023 0100 2300 0023 0000 2300 0123"
$"FF02 23FE 0423 FE0~  :FE 0323 FE03 23FE 0523 FE05 2300 0223 0002 23FF 0523 0005"
$"2300 0523 0002 230   ~23 0105 2301 0423 0104 2301 0223 0102 2302 0323 0203 2303"
$"0323 0101 2303 022   :04 2303 0123 0301 2306 0123 0200 2301 0023 0100 2301 0023"
$"0100 2300 0023 000   :00 0023 FF00 2300 0023 0000 23FD 0123 FF00 23FC 0123 FE00"
$"23FE 0023 FB00 23F~  :23 FCFF 23FE FF23 FDFF 23FE 0023 FC00 23FF 0123 FF01 2300"
$"0123 FF03 2301 012~  :03 2301 0123 0101 2303 0123 0401 2304 0023 0300 2305 FF23"
$"0BFE 2306 FE23 01~   :03 FF23 0200 2302 0023 0100 2300 0023 0102 23FF 0223 0000"
$"23FF 0223 FD03 23F   ~23 FB04 23FD 0223 FF01 23FD 0223 FE02 23FE 0223 FF02 23FF"
$"0223 FF02 2300 022   :01 2300 0223 0004 23FF 0423 FF04 23FF 0323 0001 23FF 0123"
$"FE03 23FF 0023 FFC~  :FE 0023 FC00 23FF 0023 FEFF 23FE FF23 FBFE 23FB FE23 FCFF"
$"23FD FF23 FF00 23F~  :23 FFFF 23FF 0023 FFFF 23FF FE23 0000 2300 FF23 00FF 2300"
$"FF23 00FD 2301 FC2~  :FD 2300 FE23 01FC 2300 F823 00FC 2300 FE23 00FC 23FF FA23"
$"00FB 23FF FD23 FFF~  :FE F523 FFFB 2300 FB23 00F7 2300 FB23 00FE 2301 FB23 01FB"
$"2301 FB23 01FD 230~  :23 02FC 2303 FA23 03FC 2301 FE23 02FD 2303 FC23 04FD 2303"
$"FD23 02FF 2302 FF2~  :FD 2304 FE23 04FF 2303 FF23 04FF 2308 FE23 05FF A000 A1A1"
$"00B6 0004 0001 000~  ~00 0000 0023 0000 2200 9600 D900 00A0 00A0 A100 A400 0201"
$"0101 000A 0000 000~  :00 0000 0700 0100 0109 FFFF FFFF FFFF FFFF 2200 AD00 D405"
$"E923 00E9 23F8 DC2~  ~E2 2312 DE23 20EE 2325 0323 1411 230A 1A23 021A 23F4 2423"
$"E917 23E6 1823 F11~  :00 A301 000A FFFF FFFF 011E 01FC 2200 AD00 D401 FD23 01FA"
$"2301 FB23 00FD 230~  :23 01FC 2300 FC23 00FD 2300 FF23 00FD 2300 FA23 FFF8 23FE"
$"F823 FFFB 23FF FC2~  :F7 23FF F823 00F8 2300 FC23 00FD 2302 F723 02F9 2304 F723"
$"02FC 2302 FC23 05F~  :06 F923 07FB 2304 FE23 02FF 2304 FE23 05FE 2304 FF23 04FF"
$"2303 0023 0500 230~  ~3 0200 2302 0023  501 2304 0023 0301 2305 0223 0201 2304"
$"0223 0302 2301 012~  :02 2304 0523 0405 2303 0623 0103 2301 0323 0206 2302 0823"
$"0106 2300 0323 00C~  :00 0723 FF08 23FE 0823 FE05 23FF 0423 FC08 23FC 0723 FB07"
$"23FD 0323 FA06 23F~  :23 F906 23FE 0223 FD03 23FD 0323 FC05 23FD 0323 FC06 23FD"
$"0423 FC06 23FE 03A~  :A1 A100 B600 0400 4000 80A1 00D8 0004 0000 4000 A100 D600"
$"0400 0040 00A0 00D~  :00 0000 0008 0017 23EB F9A1 00CA 0008 0000 0000 0000 0000"
$"A100 C800 0800 00C~  :00 1300 1651 00B0 00D3 00BA 0100 0700 0100 0158 A000 C9A1"
$"00D6 0004 0000 40C   :00 D7A0 00D9 A100 D600 0400 0040 00A1 00D8 0004 0000 4000"
$"A000 BEA0 00D9 09C   :~1 00A6 00AC 00D5 00BC 00FE 00AD 00D5 00AD 00D5 00AC 00D5"
$"00AC 00D7 00AC 00C~  :AC 00DC 00AC 00DD 00AD 00E1 00AE 00E5 00AF 00E8 00B0 00EB"
$"00B2 00EF 00B5 00F~  :36 00F9 00B7 00FA 00B9 00FD 00BA 00FE 00BB 00FE 00BB 00FE"
$"00BB 00FE 00BC 00F   :BC 00FB 00BC 00F8 00BC 00F7 00BC 00F5 00BC 00F2 00BB 00EE"
$"00BA 00EA 00B9 00E~  :B9 00E5 00B7 00E2 00B5 00DF 00B3 00DB 00B2 00DA 00B1 00D8"
$"00B0 00D7 00AF 00D~  :AE 00D5 00AD 00D5 7000 A600 AC00 D500 BC00 FE00 AD00 D500"
$"AD00 D500 AC00 D5C~  :00 D700 AC00 DB00 AC00 DC00 AC00 DD00 AD00 E100 AE00 E500"
$"AF00 E800 B000 EB0~  :00 EF00 B500 F600 B600 F900 B700 FA00 B900 FD00 BA00 FE00"
$"BB00 FE00 BB00 FE0~  :00 FE00 BC00 FD00 BC00 FB00 BC00 F800 BC00 F700 BC00 F500"
```

```
$"BC00 F200 BB00 EE00 EA00 EA00 B900 E800 B800 E500 B700 E200 B500 DF00 B300 DB00"
$"B200 DA00 B100 D800 B200 D700 AF00 D600 AE00 D500 AD00 D5A1 00D8 0004 0000 4000"
$"A000 BFA0 00D9 A001 07A1 00B6 0004 0001 0001 0700 0000 0022 00AD 00D5 0000 A100"
$"B600 0400 4000 8022 00DD 011D 0000 A000 A0A1 00A4 0002 0340 0988 2288 2288 2288"
$"2271 0236 000E 00F1 0170 0144 000F 0129 000F 0126 000E 0121 000E 011D 000E 0119"
$"000E 0118 000F 0115 0011 0110 0012 010E 0013 010C 0016 0107 0018 0105 0019 0103"
$"001B 0101 001D 00FF 0020 00FD 0021 00FC 0022 00FB 0026 00F9 0029 00F7 002C 00F6"
$"002F 00F5 0031 00F4 0035 00F3 0039 00F2 003D 00F2 003F 00F2 0041 00F2 0045 00F1"
$"0048 00F1 004B 00F1 004C 00F1 004D 00F1 0050 00F2 0052 00F2 0056 00F3 0057 00F3"
$"0057 00F3 0059 00F4 0059 00F4 0058 00F4 0057 00F4 0055 00F4 0051 00F4 004E 00F4"
$"004C 00F4 0048 00F4 0041 00F4 003A 00F5 0035 00F6 0033 00F6 0030 00F7 002C 00F8"
$"0028 00FA 0024 00FC 0023 00FD 0020 0100 001B 0105 0018 0108 0017 0109 0015 010D"
$"0014 010F 0013 0113 0012 0115 0011 0117 0010 011B 000F 011E 000F 0121 000F 0123"
$"000F 0126 0010 012C 0011 012E 0013 0131 0016 0135 0018 0137 0019 0138 001D 013B"
$"0022 013D 0026 013E 002A 013F 002B 013F 002E 0140 0032 0141 0034 0141 0036 0141"
$"003A 0142 003B 0142 003C 0142 003D 0142 0040 0142 0046 0141 0049 0140 004B 0140"
$"0051 013E 0054 013C 0057 013B 005C 0139 005E 0138 0060 0137 0064 0134 0066 0132"
$"0068 0131 006D 012E 006E 012D 006E 012D 006F 012C 0070 012C 006F 012C 006F 012C"
$"006D 012E 006D 012E 006C 012F 006B 0130 0069 0132 0066 0134 0063 0136 0060 0138"
$"005E 0139 005D 013A 0059 013C 0055 013E 0052 013F 004F 0140 004A 0142 0042 0144"
$"003F 0144 003E 0144 003A 0144 0035 0144 0032 0144 0030 0143 002C 0142 0025 013F"
$"0021 013D 001F 013C 001C 0139 0018 0135 0013 0130 0011 012D 0100 0A00 0000 0000"
$"0000 0007 0001 0001 2200 0F01 29F4 FE23 F502 23F7 0523 F707 23F8 0B23 FA11 23FF"
$"1023 0109 2303 0D23 FFDF 2304 EE23 0AF3 230B F723 0FFB 230D 0023 0A05 2309 0923"
$"061B 2300 0B23 FD0B 23FB 0C23 FC08 23F9 0923 FC06 230B F323 07F4 2306 EE23 02F2"
$"23FC F023 F8F2 23F4 F784 000A 0000 0000 0000 0000 A000 A301 000A FFFF FFFF 011E"
$"01FC 8400 0A00 0000 0000 0000 0022 000F 0129 FD00 23FB FF23 FC00 23FC 0023 FF00"
$"23FD 0123 FB02 23F1 0123 FE01 23FB 0323 FE02 23FE 0123 FE02 23FE 0223 FE03 23FF"
$"0123 FF01 23FE 0423 FF03 23FF 0323 FF03 23FF 0223 FF04 23FF 0423 0004 2300 0223"
$"0002 23FF 0423 0001 2300 0323 0001 2300 0123 0103 2300 0223 0104 2300 0123 0000"
$"2301 0223 0000 2300 FF23 00FF 2300 FE23 00FC 2300 FD23 00FE 2300 FC23 00F9 2301"
$"F923 01FB 2300 FE23 01FD 2301 FC23 02FC 2302 FC23 01FF 2303 FD23 05FB 2303 FD23"
$"01FF 2304 FE23 02FE 2304 FF23 02FF 2302 FF23 04FF 2303 FF23 0300 2302 0023 0300"
$"2306 0123 0201 2303 0223 0403 2302 0223 0101 2303 0423 0205 2301 0423 0104 2300"
$"0123 0103 2301 0423 0002 2300 0223 0104 2300 0123 0001 2300 0123 0003 23FF 0623"
$"FF03 2300 0223 FE04 23FE 0323 FF03 23FE 0523 FF02 23FF 0223 FD04 23FE 0223 FF02"
$"23FD 0523 FF01 2301 0023 FF01 2300 0123 00FF 2300 0023 02FE 2300 0023 01FF 2301"
$"FF23 02FE 2302 FD23 02FD 2302 FD23 01FE 2301 FF23 02FC 2302 FC23 01FD 2301 FD23"
$"02FB 2302 F823 00FF 2300 FF23 00FC 2300 FB23 00FD 23FF FE23 FFFC 23FD F923 FEFC"
$"23FF FE23 FDFD 23FC 0023 FBFB 23FD FEA0 00A1 A100 B600 0400 0100 0107 0000 0000"
$"2300 00A1 00B6 0000 0040 0080 2200 6901 3A00 00A0 00A0 A100 A400 0207 4009 AA55"
$"AA55 AA55 AA55 7101 0A00 4B00 DC00 B601 4800 6101 3F00 6301 3E00 6701 3B00 6B01"
$"3800 6F01 3400 7101 0200 7301 3000 7701 2B00 7B01 2600 8001 1F00 8201 1C00 8501"
$"1800 8901 1000 8C01 0A00 8E01 0100 8E00 FE00 8F00 FA00 8F00 F300 8F00 EE00 8E00"
$"E800 8D00 E500 8D00 E400 8C00 E200 8C00 E000 9C00 DE00 8C00 DD00 8C00 DD00 8D00"
$"DD00 9000 DD00 9000 DD00 9100 DD00 9400 DD00 9900 DD00 9C00 DD00 9F00 DC00 A000"
$"DC00 A200 DC00 A500 DC00 A500 DC00 A700 DD00 A800 DE00 AA00 E000 AB00 E200 AB00"
$"E300 AD00 E800 B101 B200 B300 F600 B400 F800 B600 FC00 B600 FD00 B600 FF00 B501"
$"0000 B501 0000 B300 0100 B001 0300 AD01 0500 AC01 0600 AB01 0700 A701 0A00 A401"
$"0D00 9F01 1200 9D01 1400 9C01 1500 9A01 1700 9801 1800 9701 1900 9601 1900 9501"
$"1800 9501 1700 9601 1600 9601 1500 9701 1300 9701 1000 9801 0C00 9801 0900 9801"
$"0800 9801 0700 9801 0500 9801 0500 9701 0500 9601 0500 9601 0500 9401 0600 9401"
$"0700 9301 0A00 9301 0A00 9301 0C00 9201 0F00 9001 1400 8E01 1900 8D01 1B00 8B01"
$"2000 8501 2A00 8101 0F00 7D01 3300 7501 3A00 7001 3D00 6E01 3E00 6A01 4000 6601"
$"4200 6001 4400 5E01 4400 5B01 4500 5801 4500 5401 4600 5001 4700 4F01 4700 4F01"
$"4700 4E01 4700 4D01 4700 4B01 4800 4C01 4800 4B01 4800 4C01 4700 4D01 4700 4D01"
$"4700 4F01 4700 5301 4400 5501 4500 5501 4500 5801 4300 5B01 4200 5F01 4000 6101"
$"3F01 000A 0000 0000 0000 0000 0700 0100 0122 0058 0143 F711 23F1 0F23 E214 23E2"
$"0423 ECF9 2303 0D23 0112 2313 0623 110A 2303 FA23 09F5 2314 EC23 EF05 23F7 0123"
$"01FB 230B FE23 16F1 0011 F123 0BEF 2304 ED23 02F4 23FE 0823 FD07 0AAA 55AA 55AA"
$"55AA 5584 000A 0000 0000 0000 A000 A301 000A FFFF FFFF 011E 01FC 8400 0A00"
$"0000 0000 0000 0002 0061 013F FF02 23FD 0423 FD04 23FC 0423 FE02 23FE 0223 FB04"
$"23FB 0423 F905 23F1 0223 FC03 23F8 0423 F903 23F8 0223 FD00 23FC 0123 F900 23FB"
$"0023 FAFF 23FD FF23 FF00 23FE FF23 FE00 23FE 0023 FF00 2300 0023 0001 2300 0323"
```

```
$"0000 2300 0123 20C    1300 0523 0003 23FF 0323 0001 2300 0223 0003 2300 0023 0102"
$"2301 0123 0202 23     123 0100 2305 0223 0A04 2304 0223 0201 2304 0223 0100 2302"
$"0023 01FF 2300 00     1FE 2302 FD23 02FD 2301 FF23 01FF 2303 FC23 03FD 2305 FB23"
$"02FE 2301 FF23 22     301 FE23 01FF 2300 FF23 FFFF 23FF 0023 FF01 23FF 0023 FE01"
$"23FD 0023 FC01 23     023 FF00 23FF 0023 FE00 2300 0023 00FF 2300 FF23 0000 2301"
$"FE23 0100 2303 FF     100 2301 0023 03FF 2305 FE23 05FE 2302 FF23 05FE 230A FA23"
$"05FC 2304 FC23 27     303 FB23 01FE 2302 FC23 02FC 2302 FA23 00FE 2301 FD23 00FD"
$"2301 FC23 01FC 23     F23 0000 2300 FF23 00FF 2301 FE23 0001 2300 FF23 FF01 2300"
$"0123 0000 2300 022    F04 23FF 0223 0000 23FE 0323 FF03 23FE 0423 FF02 A000 A1A1"
$"00B6 0004 0001 000    700 0000 0023 0000 A100 B600 0400 4000 8022 0055 00D4 0000"
$"A000 A0A1 00A4 000    140 0100 0A00 0000 0000 0000 0007 0001 0001 09FF FFFF FFFF"
$"FFFF FF22 004D 00D    008 230B 0923 12FC 2304 F923 0411 230C 0D23 1003 230F FC23"
$"07FC 23FD 0C23 060    305 01A0 00A3 0100 0AFF FFFF FF01 1E01 FC22 004D 00D2 0102"
$"2301 0423 0203 230    123 0101 2301 0123 0301 2304 0123 0300 2303 FF23 0200 2304"
$"FE23 03FF 2301 FF2    1FF 2300 0023 01FE 2300 0123 0100 2300 0023 0102 2300 0123"
$"0102 2300 0123 010    C01 0423 0204 2303 0423 0101 2302 0223 0302 2303 0223 0402"
$"2302 0023 0200 230    123 0400 2304 0023 02FF 2301 0023 03FF 2303 FF23 03FF 2301"
$"FF23 0000 2301 000    130 2300 0323 0001 2300 0123 FF04 2300 0123 0102 2301 0123"
$"0101 2302 0223 020    02 0123 0100 A000 A1A1 00B6 0004 0001 0001 0700 0000 0023"
$"0000 2200 D800 060    A0 00A0 A100 A400 020B 0109 77DD 77DD 77DD 77DD 7100 4200"
$"BE00 C600 E800 F3     C00 CF00 D800 C600 DB00 C700 DE00 CA00 E000 CE00 E200 D200"
$"E400 D500 E500 02     C0 DE00 E800 E300 E800 E800 E800 EC00 E700 EF00 CC00 F801"
$"000A 0000 0000 000    C0 0700 0100 0109 FFFF FFFF FFFF FFFF 2200 BE00 CFF7 1A23"
$"0000 2301 0323 000    303 0323 0000 2304 0223 0000 2304 0223 0000 2303 0223 0000"
$"2304 0123 0000 230    123 0000 2305 0123 0000 2305 0023 0000 2304 0023 0000 2303"
$"FF23 0000 2309 E5     00 0A77 DD77 DD77 DD77 DD84 000A 0000 0000 0000 0000 A000"
$"A301 000A FFFF FF     1E 01FC 8400 0A00 0000 0000 0000 0022 00BE 00CF F71A 2301"
$"0323 0303 2304 02     02 2303 0223 0401 2305 0223 0501 2305 0023 0400 2303 FF23"
$"09E5 A000 A1A1 00     004 0000 4000 A100 D600 0400 0040 00A0 00D7 0700 0000 0008"
$"0017 23EC FAA1 00     008 0000 0000 0000 0000 A100 C800 0800 0000 1300 1300 1651"
$"00C3 00CE 00CA 00     00 0100 0158 A000 C9A1 0CD6 0004 0000 4000 A000 D7A0 00D9"
$"A100 D600 0400 00     A1 00D8 0004 0000 4000 A000 BEA0 00D9 0800 0871 009E 00BF"
$"00CF 00CE 00F8 00     CF 00BF 00CF 00BF 00D0 00BF 00D2 00BF 00D5 00BF 00D6 00BF"
$"00D7 00C0 00DB 00     DF 00C2 00E2 00C3 00E5 00C5 00E9 00C8 00F0 00C9 00F3 00CA"
$"00F5 00CB 00F7 00     F8 00CC 00F8 00CD 00F8 00CD 00F8 00CE 00F7 00CE 00F6 00CD"
$"00F2 00CD 00F1 00     EE 00CB 00E7 00C9 00E3 00C8 00E0 00C7 00DD 00C5 00D8 00C4"
$"00D6 00C3 00D4 00     D2 00C1 00D1 00C0 00D0 00BF 00CF 00BF 00CF 7000 9E00 BF00"
$"CF00 CE00 F800 BF     F00 BF00 CF00 BF00 D000 BF00 D200 BF00 D500 BF00 D600 BF00"
$"D700 C000 DB00 C1     F00 C200 E200 C300 E500 C500 E900 C800 F000 C900 F300 CA00"
$"F500 CB00 F700 CC     800 CC00 F800 CD00 F800 CD00 F800 CE00 F700 CE00 F600 CD00"
$"F200 CD00 F100 CD     500 CB00 E700 C900 E300 C800 E000 C700 DD00 C500 D800 C400"
$"D600 C300 D400 C2     200 C100 D100 C000 D000 BF00 CF00 BF00 CFA1 00D8 0004 0000"
$"4000 A000 BFA0 00     000 D7A1 00B6 0004 0040 0080 0700 0000 0022 00BF 00C2 0000"
$"A000 A0A1 00A4 00     340 7101 AA00 C100 7B00 F300 CC00 C500 C900 C400 C80C C200"
$"C400 C100 C100 C1     800 C100 BC00 C200 BA00 C300 B800 C500 B600 C800 B300 CA00"
$"B200 CE00 B000 D4     C00 D800 A900 D900 A800 DA00 A700 DC00 A500 DC00 A400 DD00"
$"A300 DD00 A100 DD     F00 DD00 9C00 DD00 9B00 DD00 9900 DD00 9600 DD00 9200 DE00"
$"8E00 DF00 8C00 E0     A00 E200 8600 E500 8300 E800 8100 EB00 7F00 ED00 7E00 ED00"
$"7D00 EF00 7C00 F1     B00 F000 7B00 F100 7B00 F000 7C00 F000 7C00 EF00 7D00 EE00"
$"7E00 ED00 7F00 ED     F00 ED00 8000 ED00 8100 EE00 8100 EE00 8100 F000 8000 F000"
$"8000 F000 8000 F2     F00 F300 7F00 F300 7F00 F300 7F00 F200 8000 F200 8000 F000"
$"8100 F000 8100 EF     200 EC00 8400 EA00 8600 E800 8800 E700 8900 E600 8B00 E500"
$"8F00 E500 9100 E6     300 E600 9800 E600 9A00 E600 9B00 E600 9D00 E500 A000 E300"
$"A400 E200 A500 E0     F00 DB00 AD00 D800 AF00 D500 B100 D000 B400 CE00 B500 CD00"
$"B500 CC00 B600 CA     10 C900 B900 C800 BA00 C700 BB00 C600 BD00 C500 BF00 C500"
$"C000 C500 C100 C3     10 C600 C800 C700 CA00 C700 CA00 C800 CC00 C800 CB00 C800"
$"CC00 C800 CC00 C7     10 C700 CB00 C600 CA00 C500 C901 000A 0000 0000 0000 0000"
$"0700 0100 0122 00     C3 F9FA 23F4 0423 F80E 23F6 0E23 EFFD 23F2 0523 F313 230A"
$"F623 FC0C 2307 F4     FB 2308 0123 0A00 230C F823 08F4 2303 F823 07FB 2308 0023"
$"0905 23FB FB0A FF     FF FFFF FFFF 8400 0A00 0000 0000 0000 00A0 00A3 0100 0AFF"
$"FFFF FF01 1E01 FC     7A 0000 0000 0000 0000 23FF FF23 FCFE 23FD FF23 FD00 23FE"
$"0023 FE01 23FE 01     02 23FD 0323 FF02 23FE 0423 FC06 23FD 0423 FF01 23FF 0123"
$"FE02 23FF 0023 FF     FE 0023 FE00 23FD 0023 FF00 23FE 0023 FD00 23FC 0023 FC01"
$"23FE 0123 FE01 23     23 FD03 23FE 0323 FE03 23FF 0223 FF00 23FF 0223 FF02 2300"
```

328

```
$"FF23 0001 2301 FF23 0000 2301 FF23 01FF 2301 FF23 0000 2301 0023 0100 2300 0123"
$"0000 23FF 0223 0000 0000 0023 FF02 2300 0123 0000 2300 0023 01FF 2300 0023 01FE"
$"2300 0023 01FF 2302 FC23 02FE 2302 FE23 01FF 2302 FF23 04FF 2302 0023 0201 2305"
$"0023 0200 2301 0023 0200 2303 FF23 04FE 2301 FF23 03FE 2305 FB23 02FD 2302 FD23"
$"03FB 2301 FE23 00FF 0301 FF23 01FE 2302 FF23 01FF 2301 FF23 02FF 2302 FF23 0100"
$"2301 0023 0200 2305 0123 0201 2300 0023 0201 23FF 0023 0100 2300 0023 FFFF 2300"
$"0023 FFFF 23FF FFA0 0A1 A100 B600 0400 0100 0107 0000 0000 2300 00A1 00B6 0004"
$"0040 0080 2200 1201 000 00A0 00A0 A100 A400 0203 4009 8000 0800 8000 0800 7101"
$"A200 1100 F800 6F01 800 1101 2500 1101 2200 1201 1D00 1301 1800 1401 1500 1501"
$"1300 1601 1100 1801 000 1A01 0B00 1D01 0700 1F01 0500 2101 0300 2501 0000 2700"
$"FE00 2B00 FC00 2D00 000 2F00 FA00 3200 F900 3600 F800 3B00 F800 3D00 F800 4200"
$"F800 4C00 F800 5100 000 5300 F800 5700 F800 5B00 F900 5E00 FA00 6000 FB00 6200"
$"FC00 6400 FE00 6701 000 6901 0300 6A01 0400 6B01 0600 6D01 0900 6E01 0D00 6F01"
$"1000 6F01 1200 6F01 000 6F01 1800 6F01 1D00 6F01 2000 6E01 2200 6E01 2300 6C01"
$"2700 6B01 2900 6901 000 6801 2D00 6601 2F00 6101 3300 5F01 3400 5D01 3500 5701"
$"3700 5301 3800 4F01 000 4701 3B00 4401 3B00 4201 3B00 3F01 3B00 3C01 3A00 3A01"
$"3800 3901 3700 3801 000 3801 3400 3801 3200 3801 2F00 3901 2E00 3A01 2C00 3C01"
$"2700 3D01 2400 3E01 000 3E01 2000 3E01 1D00 3D01 1B00 3C01 1A00 3D01 1900 3801"
$"1700 3501 1500 3101 000 3001 1400 2F01 1400 2B01 1400 2801 1500 2601 1600 2501"
$"1700 2301 1900 2001 000 1F01 1F00 1E01 2200 1D01 2700 1D01 2A00 1D01 2B00 1C01"
$"2D00 1B01 2E00 1A01 000 1901 2F00 1801 2F00 1701 2F00 1501 2E00 1301 2C00 1201"
$"2B01 000A 0000 0000 000 0000 0700 0100 0122 0011 0125 F401 23F4 0623 F10E 23FA"
$"0D23 FF14 2301 132 00B 230C 0923 1102 230E FC23 08F8 2305 F623 04F1 2301 F123"
$"F7FA 23F8 0523 F401 3F7 F723 FEF2 2307 F823 0AFC 230B FF23 00FA 23FC FC0A 8000"
$"0800 8000 0800 8400 A00 0000 0000 0000 00A0 00A3 0100 0AFF FFFF FF01 1E01 FC84"
$"000A 0000 0000 0000 000 2200 1101 25FD 0023 FB01 23FB 0123 FD01 23FE 0123 FE01"
$"23FD 0223 FD02 23F0 323 FE02 23FE 0223 FD04 23FE 0223 FE04 23FF 0223 FF02 23FF"
$"0323 FF04 2300 0523 002 2300 0523 000A 2300 0523 0002 2300 0423 0104 2301 0323"
$"0102 2301 0223 0200 302 0323 0302 2301 0123 0201 2303 0223 0401 2303 0123 0200"
$"2302 0023 0400 2300 23 0300 2302 FF23 0100 2304 FE23 02FF 2303 FE23 01FF 2302"
$"FE23 04FB 2301 FE23 1FE 2302 FA23 01FC 2301 FC23 02F8 2300 FD23 00FE 2300 FD23"
$"FFFD 23FE FE23 FFFF FB FF23 FF00 23FE 0023 FD00 23FF 0123 FE01 23FB 0223 FD01"
$"23FE 0123 FE00 23F0 23 FEFF 23FF FF23 FFFF 23FE FD23 FEFD 23FF FC23 05FF 2300"
$"FF23 00FC 2301 FD2 1FE 2301 FF23 02FE 2304 FD23 02FF 2303 FF23 05FF 2303 0023"
$"0100 2302 FF23 01F1 301 FF23 00FF 2300 FF23 00FF 23FF FE23 FEFE 23FF FFA0 00A1"
$"A100 B600 0400 0107 .07 0000 0000 2300 0022 002B 0141 0000 A000 A0A1 00A4 0002"
$"0101 0100 0A00 000 000 0000 0007 0001 0001 09FF FFFF FFFF FFFF FF22 001B 013A"
$"0710 2300 1923 F61. F1 1423 EC17 23F0 1323 F710 23FC 09A0 00A3 0100 0AFF FFFF"
$"FF01 1E01 FC22 001 3A 0204 2302 0723 0207 2301 0823 0003 2300 0323 FF07 23FF"
$"0623 FE06 23FF 032 03 23FD 0623 FD06 23FC 0623 FE02 23FE 0323 FC05 23FC 0623"
$"FC05 23FD 0323 FB0 F7 0A23 FC05 23FE 0223 FC05 23FD 0423 FD04 23FF 0223 FE04"
$"23FC 0923 FE04 A00 07 0000 0000 2200 0E01 1D00 00A0 00A0 A100 A400 0201 0101"
$"000A 0000 0000 000 00 0700 0100 0122 000F 012A F3FF 23F1 0623 EB13 23F9 1F23"
$"051B 23FB 2323 FC1 FA 12A0 00A3 0100 0AFF FFFF FF01 1E01 FC22 000F 012A FD00"
$"23FA 0023 FB00 23F 23 FE01 23FE 0123 FC02 23FC 0323 FB04 23FE 0223 FD03 23FC"
$"0523 FC07 23FE 072 04 23FF 0423 FF07 2300 0723 0007 2301 0423 0104 2300 0723"
$"0007 23FF 0823 FF0 00 0423 FF08 23FF 0723 FF07 23FF 0423 0003 23FE 0723 FE07"
$"23FE 0823 FE05 A00 A1 00D8 0004 0000 4000 A100 D600 0400 0040 00A0 00D7 0700"
$"0000 0008 0017 23F A1 00CA 0008 0000 0000 0000 0000 A100 C800 0800 0000 1300"
$"1300 1651 00C1 00C C9 00FB 0700 0100 0109 0000 0000 0000 0000 58A0 00C9 A100"
$"D600 0400 0040 00A 07 A000 D9A1 00D6 0004 0000 4000 A100 D800 0400 0040 00A0"
$"00BE A000 D908 00C FF FFFF FFFF FFFF FF71 009E 00BD 00D0 00CD 00F9 00BE 00D0"
$"00BE 00D0 00BD 00D 5D 00D2 00BD 00D6 00BD 00D7 00BE 00DA 00C1 00E1 00C2 00E5"
$"00C3 00E8 00C4 00E C5 00EF 00C7 00F3 00C7 00F4 00C8 00F6 00C9 00F7 00CA 00F8"
$"00CC 00F9 00CC 00F CC 00F9 00CD 00F7 00CD 00F6 00CC 00F3 00CC 00F2 00CC 00F1"
$"00CB 00ED 00CA 00E C9 00E6 00C8 00E3 00C7 00DF 00C4 00D8 00C2 00D5 00C1 00D3"
$"00C0 00D1 00BF 00D BF 00D0 00BE 00D0 0900 0000 0000 0000 0070 009E 00BD 00D0"
$"00CD 00F9 00BE 00D BE 00D0 00BD 00D0 00BD 00D2 00BD 00D6 00BD 00D7 00BE 00DA"
$"00C1 00E1 00C2 00E C3 00E8 00C4 00EB 00C5 00EF 00C7 00F3 00C7 00F4 00C8 00F6"
$"00C9 00F7 00CA 00F CC 00F9 00CC 00F9 00CC 00F9 00CD 00F7 00CD 00F6 00CC 00F3"
$"00CC 00F2 00CC 00F CB 00ED 00CA 00E9 00C9 00E6 00C8 00E3 00C7 00DF 00C4 00D8"
$"00C2 00D5 00C1 00D C0 00D1 00BF 00D0 00BF 00D0 00BE 00D0 A100 D800 0400 0040"
$"00A0 00BF A000 D9A0 07 A100 B600 0400 4000 8007 0000 0000 2200 CF00 BD00 00A0"
$"00A0 A100 A400 0201 01 000A 0000 0000 0000 0000 0700 0100 0109 FFFF FFFF FFFF"
```

```
$"FFFF 2200 0E00 00"    23 0210 23F9 0623 F6FC 23EC F423 DEFB 23F6 0E23 020E 23F7"
$"0423 F901 23FC 05"    FD 23F9 07A0 00A3 0100 0AFF FFFF FF01 1E01 FC22 00CE 00C9"
$"FF00 23FC 0123 FF"    FE 0123 FF01 23FF 0123 FF00 2300 0123 0001 2300 0223 0004"
$"2300 0223 FF02 23"    23 FF01 23FD 0123 FE00 23FF 0023 FEFF 23FF 0023 FDFE 23FC"
$"FE23 FCFE 23FD FE"    FF 23FA FD23 F9FE 23F9 FF23 FBFF 23FE 0023 FCFF 23FC 0023"
$"FE00 23FC 0123 FF"    FE 0123 FF01 23FF 0123 FF01 23FE 0423 FF04 2300 0323 0002"
$"2300 0123 0004 23"    23 FF02 23FF 0023 FE01 23FB 0223 FF00 23FF 0023 FF00 23FE"
$"0123 FF01 23FF 01"    00 23FE 0223 FE00 23FD 0023 FF0 2300 0023 FE00 23FD 0023"
$"FE00 23FF 0123 FF"    FD 0223 FF01 23FF 01A0 00A1 A100 B600 0400 0100 0107 0000"
$"0000 2300 00A0 00"    100 B600 0400 0C00 8007 0001 0001 2000 0801 0000 0800 22A1"
$"00B6 0004 0001 00"    FF0 0000 0023 0000 A100 B600 0400 0C00 8007 0001 0001 2000"
$"1100 EA00 1100 30"    B6 0004 0001 0001 0700 0000 0023 0000 A100 B600 0400 0C00"
$"8007 0001 0001 22"    5F0 DF00 1B00 41A1 00B6 0004 0001 0001 0700 0000 0023 0000"
$"A100 B600 0400 0C"    007 0001 0001 2000 2500 D800 2500 23A1 00B6 0004 0001 0001"
$"0700 0000 0023 00"    100 B600 0400 0C00 8007 0001 0001 2200 3200 D28B 00A1 00B6"
$"0004 0001 0001 07"    100 0023 0000 A100 B600 0400 0C00 8007 0001 0001 2200 3E00"
$"CD9D 00A1 00B6 00"    001 0001 0700 0000 0023 0000 A100 B600 0400 0C00 8007 0001"
$"0001 2200 4900 CA"    A1 00B6 0004 0001 0001 0700 0000 0023 0000 A100 B600 0400"
$"0C00 8007 0001 00"    100 5400 CACA 00A1 00B6 0004 0001 0001 0700 0000 0023 0000"
$"A100 B600 0400 0C"    007 0001 0001 2200 5E00 CA84 00A1 00B6 0004 0001 0001 0700"
$"0000 0023 0000 A1"    500 0400 0C00 8007 0001 0001 2200 6900 CDC2 00A1 00B6 0004"
$"0001 0001 0700 00"    23 0000 A100 B600 0400 0C00 8007 0001 0001 2200 7300 D0D3"
$"00A1 00B6 0004 00"    01 0700 0000 0023 0000 A100 B600 0400 0C00 8007 0001 0001"
$"2200 7C00 D1CE 00"    B6 0004 0001 0001 0700 0000 0023 0000 A100 B600 0400 0C00"
$"8007 0001 0001 22"    00 D2AB 00A1 00B6 0004 0001 0001 0700 0000 0023 0000 A100"
$"B600 0400 0C00 30"    001 0001 2200 9400 D3DA 00A1 00B6 0004 0001 0001 0700 0000"
$"0023 0000 A100 B6"    400 0C00 8007 0001 0001 2200 9E00 D0ED 00A1 00B6 0004 0001"
$"0001 0700 0000 00"    00 A100 B600 0400 0C00 8007 0001 0001 2200 A800 CEE8 00A1"
$"00B6 0004 0001 00"    00 0000 0023 0000 A000 8DA1 0096 000C 0300 0000 0200 0000"
$"0000 0000 A100 9A"    500 0400 0000 2D00 0001 000A 00BB 00D1 00EB 01E3 2C00 0E00"
$"210B 4176 616E 74"    61 7264 6503 0021 0401 0D00 1828 00CE 012C 0E14 5249 4654"
$"2050 4154 5445 52"    04 032B 3518 0E53 696D 756C 6174 696F 6E20 2020 20A0 0097"
$"0100 0AFF FFFF FF"    E01 FC07 0001 0001 2000 B601 0500 B601 E1A1 00B6 0004 0040"
$"0080 0700 0000 00"    C9 00C0 0000 A000 A0A1 00A4 0002 0140 0100 0A00 0000 0000"
$"0000 0007 0001 00"    200 C800 CCF4 0123 0210 23F9 0623 F6FC 23EC F423 DEFB 23F6"
$"0E23 020E 23F7 04"    A01 23FC 0623 F500 23ED FF23 F30D A000 A301 000A FFFF FFFF"
$"011E 01FC 2200 C8"    FF 0023 FC01 23FF 0023 FE01 23FF 0123 FF01 23FF 0323 0001"
$"2300 0123 0002 23"    23 0002 23FF 0223 FF01 23FF 0123 FD01 23FE 0023 FF00 23FE"
$"FF23 FF00 23FD FE"    FE 23FC FE23 FDFE 23FE FF23 FAFD 23F9 FE23 F9FF 23FB FF23"
$"FE00 23FC FF23 FC"    FE 0023 FC01 23FE 0123 FE01 23FF 0123 FF01 23FF 0123 FE04"
$"23FF 0423 0003 23"    23 0001 2300 0423 FF02 23FF 0223 FF00 23FE 0123 FB02 23FF"
$"0023 FF00 23FF 00"    01 23FF 0123 FF01 2300 0023 FE02 23FE 0123 FE00 23FF 0023"
$"FF00 23FD 0023 FC"    FB FF23 FE00 23FF 0023 FE00 23FD 0123 FD01 23FD 0223 FD02"
$"23FC 0323 FE02 23F"    A0 00A1 A100 B600 0400 0100 0107 0000 0000 2300 00A0 008C"
$"A100 B600 0400 40"    07 0001 0001 0900 0000 0000 0000 0020 00D5 01E8 00D5 0147"
$"A100 B600 0400 01"    07 0000 0000 2300 00A1 00B6 0004 0040 0080 0700 0100 0120"
$"00D8 01E8 00D8 01"    00 B600 0400 0100 0107 0000 0000 2300 00A1 00B6 0004 0040"
$"0080 0700 0100 01"    DB 01E8 00DB 0147 A100 B600 0400 0100 0107 0000 0000 2300"
$"00A1 00B6 0004 00"    80 0700 0100 0120 00DE 01E8 00DE 0147 A100 B600 0400 0100"
$"0107 0000 0000 23"    A1 00B6 0004 0040 0080 0700 0100 0120 00E1 01E8 00E1 0147"
$"A100 B600 0400 01"    07 0000 0000 2300 00A1 00B6 0004 0040 0080 0700 0100 0120"
$"00E4 01E8 00E4 01"    00 B600 0400 0100 0107 0000 0000 2300 00A1 00B6 0004 0040"
$"0080 0700 0100 01"    E6 01E8 00E6 0147 A100 B600 0400 0100 0107 0000 0000 2300"
$"00A1 00B6 0004 00"    30 0700 0100 0120 00E9 01E8 00E9 0147 A100 B600 0400 0100"
$"0107 0000 0000 23"    A0 008D A100 D800 0400 0040 00A1 00D6 00 1 00C0 4000 AC00"
$"D708 0017 23A2 CD"    CA 0008 0000 0000 0000 0000 A100 C800 0800 0000 1300 1300"
$"1609 FFFF FFFF FF"    FF 5100 B100 D300 BB00 FF07 0001 0001 0900 0000 0000 0000"
$"0058 A000 C9A1 00"    14 0000 4000 A000 D7A0 00D9 A100 D600 0400 0040 00A1 00D8"
$"0004 0000 4000 A0"    A0 00D9 0800 0809 FFFF FFFF FFFF FFFF 7100 9E00 AE00 D400"
$"BE00 FD00 AF00 D4"    00 D400 AE00 D500 AE00 D700 AE00 DA00 AE00 DB00 AF00 DF00"
$"B100 E600 B200 EA"    00 ED00 B400 F000 B600 F400 B700 F600 B800 F800 B900 FA00"
$"BA00 FB00 BB00 F0"    00 FD00 BD00 FD00 BE00 FD00 BE00 FD00 BE00 FF00 BE00 F700"
$"BE00 F600 BE00 F5"    00 F100 BC00 ED00 BB00 EA00 BA00 E700 B900 E300 B600 DC00"
$"B400 D900 B300 D8"    00 D500 B000 D400 B000 D400 AF00 D409 0000 0000 0000 0000"
```

330

$"7200 9E00 AE00 D400 BE00 FD00 AF00 D400 AF00 D400 AE00 D500 AE00 D700 AE00 DA00"
$"AE00 DB00 AF00 DF00 B100 E600 B200 EA00 B300 ED00 B400 F000 B600 F400 B700 F600"
$"B800 F900 B900 FA00 BA00 FB00 BB00 FC00 BC00 FD00 BD00 FD00 BE00 FD00 BE00 FD00"
$"BE00 FB00 BE00 F700 BE00 F600 BE00 F500 BD00 F100 BC00 ED00 BB00 EA00 BA00 E700"
$"B900 E300 B600 DC00 B400 D900 B300 D800 B100 D500 B000 D400 B000 D400 AF00 D4A1"
$"00C8 0004 0000 4000 A000 BFA0 00D9 A000 D7A1 00B6 0004 0040 0080 0700 0000 0022"
$"00DB 00E0 0000 A000 A0A1 00A4 0002 0140 0100 0A00 0000 0000 0000 0007 0001 0001"
$"00FF FFFF FFFF FFFF FF02 0058 00D8 0823 2300 1923 FE12 23FD 1023 FE06 23FD 06A0"
$"00A3 0100 0AFF FFFF FF01 1E01 FC22 0058 00D8 0104 2302 0923 0106 2302 0823 0106"
$"2300 0423 0106 2300 0323 0002 2300 0323 0005 2300 0523 FF06 2300 0223 0004 23FE"
$"0923 FF04 2300 0223 FF04 2300 0223 FF03 2300 0023 0000 23FF 0323 FF03 23FF 0323"
$"FF02 A000 A1A1 00B6 0004 0001 0700 0000 0023 0000 A100 B600 0400 4000 9022"
$"0093 00FA 0000 A000 A0A1 00A4 0002 0140 0100 0A00 0000 0000 0000 0007 0001 0001"
$"2200 7201 0AF0 2123 F51B 23F9 1323 FE06 A000 A300 000A FFFF FFFF 011E 01FC 2200"
$"7201 0AFE 0423 FC08 23FD 0723 FD07 23FD 0623 FE05 23FE 0523 FF04 23FF 0223 FF03"
$"23FD 0723 FE06 23FE 0523 FF03 23FF 0323 FE04 23FF 0423 FF03 23FF 02A0 00A1 A100"
$"B600 0400 0100 0107 0000 00C0 2300 00A1 00B6 0004 0040 0080 2200 9101 1700 00A0"
$"00A0 A100 A400 0201 0001 000A 0000 0000 0000 0000 0700 0100 0122 007E 012C EB13"
$"2400 1723 F21B 23FD 0AA0 00A3 0100 0AFF FFFF FF01 1E01 FC22 007E 012C FE02 23FA"
$"0023 FC04 23FC 0423 FB04 23FE 0323 FC04 23FD 0323 FF01 23FE 0323 FB06 23FC 0623"
$"FB06 23FE 0423 FE04 23FD 0623 FE06 23FE 0523 FF03 A000 A1A1 00B6 0004 0001 0001"
$"0700 0000 0023 0000 A100 B600 0400 4000 8022 00B8 00D4 0000 A000 A0A1 00A4 0002"
$"0140 0100 0A00 0000 0000 0000 0007 0001 0001 2200 BF00 D202 F923 01FB 2300 FCA0"
$"00A3 0100 0AFF FFFF FF01 1E01 FC22 00BF 00D2 01FE 2300 FE23 01FE 2301 FD23 0000"
$"2300 FF23 00FC 2300 0AA0 00A1 A100 B600 0400 0100 0107 0000 0000 2300 00A1 00B6"
$"0004 0040 0080 2200 00C0 E300 00A0 00A0 A100 A400 0201 4001 000A 0000 0000 0000"
$"0000 0700 0100 0122 00C2 00E1 02FA 2301 FAA0 00A3 0100 0AFF FFFF FF01 1E01 FC22"
$"00C2 00E1 01FD 2302 0A23 00FD A000 A1A1 00B6 0004 0001 0001 0700 0000 0023 0000"
$"A100 B600 0400 4000 0022 00C6 00F8 0000 A000 A0A1 00A4 0002 0140 0100 0A00 0000"
$"0000 0000 0007 0001 0001 2200 CA00 F602 FC23 02FB 2303 FCA0 00A3 0100 0AFF FFFF"
$"FF01 1E01 FC22 00CA 00F6 01FE 2302 FC23 00FF 2301 FF23 02FD 2301 FEA0 00A1 A100"
$"B600 0400 0100 0100 0000 0000 2300 00A1 00B6 0004 0040 0080 2200 C800 E700 00A0"
$"00A0 A100 A400 0201 0001 000A 0000 0000 0000 0000 0700 0100 0109 0000 0000 0000"
$"0000 2200 C400 E8FF 0423 0000 23FF 0123 0000 A000 A301 000A FFFF FFFF 011E 01FC"
$"2200 C400 E9FF 0401 0001 A000 A1A1 00B6 0004 0001 0001 0700 0000 0023 0000 A100"
$"B600 0400 4000 8022 00CA 00F3 0000 A000 A0A1 00A4 0002 0940 0100 0A00 0000 0000"
$"0000 0007 0001 0001 2200 C800 F300 0223 0000 23FF 0123 0000 2300 0123 0000 2300"
$"0023 0000 A000 A301 000A FFFF FFFF 011E 01FC 2200 C800 F300 0223 FF01 2300 0123"
$"0000 A000 A1A1 00B6 0004 0001 0001 0700 0000 0023 0000 A100 B600 0400 4000 8007"
$"0001 0001 2200 BF00 08FF 04A1 00B6 0004 0001 0001 0700 0000 0023 0000 A100 B600"
$"0400 4000 8022 00D0 00A2 0000 A000 A0A1 00A4 0002 0140 0100 0A00 0000 0000 0000"
$"0007 0001 0001 2200 B000 A6FC 0423 FD01 23FC 0123 FBFF 23FB FF23 F901 23FC 0323"
$"FB03 A000 A301 000A FFFF FFFF 011E 01FC 4200 DB00 A6FF 0123 FE02 23FF 0123 FF00"
$"2300 0023 FF01 23FD 0123 FF00 23FF 0023 FE00 23FF 0023 FF00 23FD FF23 FF00 23FF"
$"0023 FC00 23FE 0023 FF00 23FD 0123 FF01 2300 0023 FE01 23FE 0123 FF01 23FE 01A0"
$"00A1 A100 B600 0400 0100 0107 0000 0000 2300 00A1 0096 0000 0200 0000 0500 0000"
$"0000 0002 A100 9A00 08FF F800 0000 7C00 0001 000A 008D 00DA 00BE 01FB 2C00 0800"
$"1405 5469 6D65 7303 0C14 0419 0D00 2828 00AD 00EE 0742 414C 4C4F 4F4E A000 97A1"
$"00B6 0004 000C 0080 A100 B600 0400 0100 0101 000A FFFF FFFF 011E 01FC 2200 AD01"
$"E600 00A1 00B6 0004 000C 0080 A100 B600 0400 0100 0123 0000 A100 B600 0400 4000"
$"80A1 00B6 0004 000C 0001 2300 00A1 00B6 0004 0040 0080 A100 B600 0400 0100 0123"
$"0000 A100 B600 0400 0000 80A1 00B6 0004 0001 0001 2300 00A1 00B6 0004 0040 0080"
$"A100 B600 0400 0100 0123 0000 A100 B600 0400 4000 80A1 00B6 0004 0001 0001 2300"
$"00A1 00B6 0004 000C 0080 A100 B600 0400 0100 0123 0000 A100 B600 0400 4000 80A1"
$"00B6 0004 0001 0001 2300 00A1 00B6 0004 0040 0080 A100 B600 0400 0100 0123 0000"
$"A100 B600 0400 4000 00A1 00B6 0004 0001 0001 2300 00A1 00B6 0004 000C 0080 A100"
$"B600 0400 0100 0123 0000 A100 B600 0400 0C00 80A1 00B6 0004 0001 0001 2300 00A1"
$"00B6 0004 000C 0000 00B6 0004 0100 0123 0000 A100 B600 0400 0C00 80A1 00B6 0004"
$"0004 0001 0001 23FF 0AA1 00B6 0004 000C 0080 A100 B600 0400 0100 0123 0000 A100"
$"B600 0400 0C00 80A1 00B6 0004 0001 0001 2300 00A1 00B6 0004 000C 0080 A100 B600"
$"0400 0100 0123 0000 00B6 0400 0400 0C00 80A1 00B6 0004 0001 0001 2300 00A1 00B6"
$"0004 000C 0080 A100 0000 0400 0100 0123 0000 A100 B600 0400 0C00 80A1 00B6 0004"
$"0001 0001 2300 00A1 00B6 0004 000C 0080 A100 B600 0400 0100 0123 0000 A100 B600"
$"0400 0C00 80A1 00A1 00B4 0001 0001 2300 00A1 00B6 0004 000C 0080 A100 B600 0400"

```
        $"0100 0123 0000 A100  000 0400 0C00 80A1 00B6 0004 0001 0001 2300 00A1 00B6 0004"
        $"0000 0080 A100 B600  000 0100 0123 0000 A100 B600 0400 0C00 80A1 00B6 0004 0001"
        $"0001 2300 00A1 00B6  004 0040 0080 A100 B600 0400 0100 0123 0000 A100 B600 0400"
        $"4000 80A1 00B6 0004  001 0001 2300 00A1 00B6 0004 0040 0080 A100 B600 0400 0100"
        $"0123 0000 A100 B600  000 4000 80A1 00B6 0004 0001 0001 2300 00A1 00B6 0004 0040"
        $"0080 A100 B600 0400  000 0123 0000 A100 B600 0400 4000 80A1 00B6 0004 0001 0001"
        $"2300 00A1 00B6 0004  040 0080 A100 B600 0400 0100 0123 0000 A100 B600 0400 4000"
        $"80A1 00B6 0004 0001  001 2300 00A1 00B6 0004 0040 0080 A100 B600 0400 0100 0123"
        $"0000 A100 B600 0400 4000 80A1 00B6 0004 0001 0001 2300 00A1 00B6 0004 0040 0080"
        $"A100 B600 0400 0100 0123 0000 A100 B600 0400 4000 80A1 00B6 0004 0001 0001 2300"
        $"00A1 00B6 0004 0040 0080 A100 B600 0400 0100 0123 0000 A100 B600 0400 4000 80A1"
        $"00B6 0004 0001 0001 2300 00A1 00B6 0004 0040 0080 A100 B600 0400 0100 0123 0000"
        $"A100 B600 0400 4000 80A1 00B6 0004 0001 0001 2300 00A1 00B6 0004 0040 0080 A100"
        $"B600 0400 0100 0123 0000 A100 B600 0400 4000 80A1 00B6 0004 0001 0001 2300 00A1"
        $"00B6 0004 0040 0080 A100 B600 0400 0100 0123 0000 A000 83FF"
};

resource 'CNTL' (128, "scroll thing") {
        {0, 155, 234, 170},
        0,
        visible,
        0,
        0,
        scrollBarProc,
        0,
        "scroll thing"
};

data 'CDEF' (133, "Popup menu") {
        $"600E 0000 4344 4546 0085 0000 0000 0000"          /* `...CDEF.Ö...... */
        $"41FA FFEE 21C8 09CE 6000 08AC 48E7 C0C0"          /* A...!»ΔŒ`..`H.¿¿ */
        $"322F 0014 206F 0016 2248 7000 22C0 22C0"          /* 2/.. o.."Hp."¿"¿ */
        $"22C0 22C0 22C0 22C0 22C0 22C0 226F 001A"          /* "¿"¿"¿"¿"¿"¿"o.. */
        $"5341 671E 701F 9041 E249 41F0 0000 30D9"          /* SAg.p.êA.IA...0. */
        $"51C9 FFFC 4CDF 0303 2F57 000A 4FEF 000A"          /* Q....L.../W..O... */
        $"4E75 1159 001F 60EC 2F0A 226F 0008 246F"          /* Nu.Y..`./."o..$o */
        $"000C 302F 0010 3400 121A B202 6402 1401"          /* ..0/..4...≤.d... */
        $"12C2 6002 12DA 51CA FFFC B001 245F 205F"          /* .¬`...Q ..∞.$_ _ */
        $"4FEF 000A 4ED0 225F 201F A04C 2E80 7000"          /* O...N-"_ .†L.Äp. */
        $"2F09 31C0 0220 4E75 7000 60F6 225F 201F"          /* /Δ1¿. Nup.`."_ . */
        $"A122 2E88 4EFA FFEA 225F 205F A023 4EFA"          /* °".àN..."_ †$N. */
        $"FFE0 225F 205F A029 4EFA FFD6 225F 205F"          /* .."_ †)N..+"_ _ */
        $"A02A 4EFA FFCC 4E56 FFC6 206E 000C 43EE"          /* †*N..ÄNV.Δ n..C. */
        $"FFE8 22D8 22D8 1D7C 0055 FFCE 1D7C 00AA"          /* .."ÿ"ÿ.|.Ü.Œ.|.™ */
        $"FFCF 1D7C 0055 FFD0 1D7C 00AA FFD1 1D7C"          /* .œ.|.Ü.-.|.™.-.| */
        $"0055 FFD2 1D7C 00AA FFD3 1D7C 0055 FFD4"          /* .Ü.".|.™.".|.Ü.` */
        $"1D7C 00AA FFD5 486E FFD6 A898 A89E 486E"          /* .|.™.'Hn.+®ô®ÛHn */
        $"FFCE A89D 3F3C 000B A89C 486E FFE8 A8A2"          /* .Œ®ù?<..®úHn..®¢ */
        $"486E FFD6 A899 4E5E 205F 504F 4ED0 4E56"          /* Hn.+®ôN^ _PON-NV */
        $"FDE8 48E7 0308 7E01 4246 422E FDEC 594F"          /* ..H...~.BFB...YO */
        $"3F2E 0010 A9BF 285F 200C 6748 2F0C 3F07"          /* ?...©ø(_ .gH/.?. */
        $"486E FEEC A946 554F 486E FEEC A88C BC5F"          /* Hn..©FUOHn..®å°_ */
        $"6C1A 3F3C 00FF 486E FEEC 486E FDEC 4EBA"          /* 1.?<..Hn..Hn..N∫ */
        $"FEE8 554F 486E FEEC A88C 3C1F 5247 0C47"          /* ..UOHn..®å<.RG.G */
        $"0064 5EC0 7200 122E FEEC 4A41 57C1 8001"          /* .d^¿r.....JAW¡Ä. */
        $"67BA 6016 41FA 0038 43EE FDEC 22D8 22D8"          /* g∫`.A..8C..."ÿ"ÿ */
        $"554F 486E FEEC A88C 3C1F 3D46 0012 206E"          /* UOHn..®å<.=F.. n */
        $"000C 3F3C 00FF 486E FDEC 2F08 4EBA FE9A"          /* ..?<..Hn../.N∫.ö */
        $"4CDF 10C0 4E5E 205F 4FEF 000A 4ED0 043F"          /* L..¿N^ _O...N-.? */
        $"3F3F 3F08 4E56 FEEC 2F0C 594F 3F2E 000E"          /* ???.NV../.YO?... */
        $"A9BF 285F 200C 6742 2F0C 3F2E 000C 486E"          /* ©ø(_ .gB/.?...Hn */
        $"FEEC A946 7000 102E FEEC 4A40 6F08 486E"          /* ..©Fp...J@o.Hn */
        $"FEEC A884 602A 2F0C 3F3C 0001 486E FEEC"          /* ..®Ñ`*..?<..Hn.. */
        $"A946 206E 0008 2068 0010 2050 317C 0001"          /* ©F n.. h.. P1!.. */
```

332

```
$"0012 486E FEEC A884 6005 487A 000E A884"    /* ..Hn..ÐÑ`.Hz..ÐÑ */
$"285F 4E5E 205F 504F 4ED0 033F 3F3F 4E56"    /* (_N^ _PON-.???NV */
$"FB46 48E7 1F38 286E 0008 3D7C FFE0 FB78"    /* .FH..8(n..=|...x */
$"3D7C 7FC0 FB7A 3D7C 3580 FB7C 3D7C 1F00"    /* =|.¿.z=|?Ä.|=|.. */
$"FB7E 3D7C 0E00 FB80 3D7C 0400 FB82 2F2E"    /* .~=|...Ä=|...Ç/. */
$"0010 4EBA FE3E 206E 0010 2050 7000 1028"    /* ..Nʃ.> n.. Pp..( */
$"0010 4A40 6700 03B4 206E 0010 2050 41E8"    /* ..J@g..¥ n.. PA. */
$"0008 43EE FFE8 22D8 22D8 486E FFBE A898"    /* ..C..."ÿ"ÿHn.æÐò */
$"A89E 486E FFB6 A874 206E FFB6 3628 0044"    /* ÐûHn.ðÐt n.ð6(.D */
$"206E FFB6 3828 004A 206E FFB6 3A28 0048"    /* n.ð8(.J n.ð:(.H */
$"206E FFB6 4868 0046 486E FB48 3F3C 0001"    /* n.ðBh.FHn.H?<.. */
$"4EBA FD3A 1D6E FB67 FFAF 4267 A887 3F3C"    /* Nʃ.:.n.g.ØBgÐá?< */
$"000C A88A 3F3C 0001 A889 422E FB66 7000"    /* ..Ðä?<..ÐaB..fp. */
$"102E FB66 3F00 A888 486C FFE4 A88B 594F"    /* ...f?.ÐaHl..ÐaYO */
$"A8D8 2E1F 2F07 A87A 594F A8D8 245F 2F0A"    /* Ðy.././.ÐzYOÐÿS_/. */
$"486E FFE8 A8DF 2F07 2F0A 2F0A A8E4 2F0A"    /* Hn..Ð./././.Ð./. */
$"A879 486E FFE8 A8A3 206E 0010 2050 4AA8"    /* ÐyHn..Ð£ n.. PJÐ */
$"001C 6600 0162 594F 7012 2F00 4EBA FD5E"    /* ..f..bYOp./.Nʃ.^ */
$"295F FFEC 206E 0010 2050 216C FFEC 001C"    /* )_.. n.. P!l.... */
$"2F2C FFEC 4EBA FD5C 206C FFEC 2650 4253"    /* /,..Nʃ.\ l..&PBS */
$"426B 0002 426B 0004 426B 0006 426B 0008"    /* Bk..Bk..Bk..Bk.. */
$"41EE FFE8 43EB 000A 22D8 22D8 594F 2F3C"    /* A...C..."ÿ"ÿYO/< */
$"4D45 4E55 206E 0010 2050 3F28 0016 A9A0"    /* MENU n.. P?(..Ct */
$"2C1F 4A86 6712 2F06 486E FCA6 486E FCA2"    /* ,.J@g./.Hn.¶Hn.¢ */
$"486E FCAC A9A8 600C 41FA 02A6 43EE FCAC"    /* Hn.¬C®`.A..¶C..¬ */
$"22D8 22D8 206E 0010 2050 3F3C 00FF 486E"    /* "ÿ"ÿ n.. P?<..Hn */
$"FCAC 4868 0028 4EBA FC90 554F 486E FCAC"    /* .¬Hh.(Nʃ.âUOHn.¬ */
$"A88C 206C FFEC 2050 309F 206C FFEC 2050"    /* Ðä l.. POü l.. P */
$"4A50 6F0C 206C FFEC 2050 317C 0005 0002"    /* JPo. l.. Pl|.... */
$"302C FFE4 5240 5240 206C FFEC 2050 3140"    /* 0,..R@R@ l.. P1@ */
$"0004 554F 206E 0010 2050 3F28 0016 486E"    /* ..UO n.. P?(..Hn */
$"FBA2 2F0E 4EBA FD18 3D5F FFE0 486E FFD0"    /* .¢/.Nʃ..=_..Hn.- */
$"4267 4267 302E FFE0 0640 000D 5A40 3F00"    /* BgBg0....@.¬Z@?. */
$"302C FFE4 D06C FFE6 5440 3F00 A8A7 486E"    /* 0,..-l..T@?.Ð£Hn */
$"FFD0 206C FFEC 2050 226C FFEC 2251 3010"    /* .- l.. P"l.."Q0. */
$"D069 0002 5640 3F00 3F3C 0001 A8A8 206C"    /* -i..V@?.?<..ÐÐ l */
$"FFEC 2050 43EE FFD0 41E8 000A 20D9 20D9"    /* .. PC..-A... . . */
$"302E FFD2 0640 000D 206C FFEC 2050 3140"    /* 0.."@.¬ l.. P1@ */
$"0006 302E FFD0 D06C FFE4 5240 206C FFEC"    /* ..0..--l..R@ l.. */
$"2050 3140 0008 206E 0010 2050 2968 001C"    /* P1@.. n.. P)h.. */
$"FFEC 2F2C FFEC 4EBA FC0A 206C FFEC 2050"    /* ../,..Nʃ.. l.. P */
$"302E FFEA D068 0002 3F00 206C FFEC 2050"    /* 0...-h..?. l.. P */
$"302E FFE8 D068 0004 3F00 A893 206E 0010"    /* 0...-h..?.Ði n.. */
$"2050 7000 1028 0028 4A40 6F0C 206E 0010"    /* Pp..(.(J@o. n.. */
$"2050 4868 0028 A884 206C FFEC 2050 41E8"    /* PHh.(ÐÑ l.. PA. */
$"000A 43EE FFD0 22D8 22D8 486E FFD0 3F2E"    /* ..C..-"ÿ"ÿHn.-?. */
$"FFEA 3F2E FFE8 A8A8 206C FFEC 2050 302E"    /* ..?...ÐÐ l.. PO. */
$"FFEA D068 0006 3F00 206C FFEC 2050 302E"    /* ..-h..?. l.. PO. */
$"FFE8 D068 0008 3F00 A893 206E 0010 2050"    /* ..-h..?.Ði n.. P */
$"3F28 0016 206E 0010 2050 3F28 0012 2F0E"    /* ?(.. n.. P?(../. */
$"4EBA FC92 486E FFD0 A8A1 302E FFD2 5240"    /* Nʃ.iHn.-Ð¡0.-®°0.."R@ */
$"3F00 3F2E FFD4 A893 3F2E FFD6 3F2E FFD4"    /* ?.?..`Ði?..+?..` */
$"A891 3F2E FFD6 302E FFD0 5240 3F00 A891"    /* Ðö?..+0..-R@?.Ðö */
$"486E FB8C 4267 4267 3F3C 0010 3F3C 0006"    /* Hn.àBgBg?<..?<.. */
$"A8A7 41EE FB78 2D48 FB94 3D7C 0002 FB98"    /* ÐßA..x-H.1=|...ò */
$"41EE FB8C 43EE FB9A 22D8 22D8 41EE FB8C"    /* A..àC..ö"ÿ"ÿA..à */
$"43EE FB84 22D8 22D8 486E FB84 302E FFD6"    /* C..Ñ"ÿ"ÿHn.ÑO..+ */
$"0640 FFEF 3F00 302E FFD0 5C40 3F00 A8A8"    /* .@..?.0..-\@?.ÐÐ */
$"486E FFBA A874 486E FB94 206E FFBA 4868"    /* Hn.ʃÐtHn.1 n.ʃHh */
$"0002 486E FB8C 486E FB84 4267 42A7 A8EC"    /* ..Hn.àHn.ÑBgBßÐ®. */
$"206E 0010 2050 7000 1028 0011 0C40 00FF"    /* n.. Pp..(...@.. */
$"660A 486E FFE8 2F0E 4EBA FACC 2F07 A879"    /* f.Hn../.Nʃ.Ã/.Ðy */
$"2F07 A8D9 2F0A A8D9 3F03 A887 3F04 A88A"    /* /.Ð./.Ð.?.Ðá?.Ðä */
$"3F05 A889 486E FFAF 486E FB48 3F3C 0001"    /* ?.ÐâHn.ØHn.H?<.. */
```

```
    $"4EBA F9EA 1D6E FB67 FB46 7000 102E FB46"      /* N∫...n.g.Fp....F */
    $"3F00 A888 486E FFBE A899 2F2E 0010 4EBA"      /* ?.®àHn.æ®ô/...N∫ */
    $"FA7C 4CDF 1CF9 4E5E 205F 4FEF 000E 4ED0"      /* .|L...N^ _O...N- */
    $"043F 3F3F 3F08 4E56 FFE8 2F0C 286E 0010"      /* .????.NV.../.(n.. */
    $"42AE 0016 2F0C 4EBA FA4A 2054 7000 1028"      /* B®.../.N∫.J Tp..( */
    $"0011 0C40 00FF 6762 2054 7000 1028 0011"      /* ...@..gb Tp..(.. */
    $"0C40 00FE 674C 2054 41E8 0008 43EE FFE8"      /* .@..gL TA...C... */
    $"22D8 22D8 302E FFEE 5B40 3D40 FFEA 302E"      /* "ÿ"ÿ0...[@=@..0. */
    $"FFEC 5B40 3D40 FFE8 554F 2F2E 000C 2054"      /* ..[@=@..UO/... T */
    $"4868 0008 A8AD 554F 2F2E 000C 486E FFE8"      /* Hh..®≠UO/...Hn.. */
    $"A8AD 101F 5300 C01F 6710 700A 2D40 0016"      /* ®≠..S.¿.g.p.-@.. */
    $"6008 2D7C 0000 00FE 0016 2F0C 4EBA F9DE"      /* `.-|....../.N∫.. */
    $"285F 4E5E 205F 4FEF 000E 4ED0 4E56 FF78"      /* (_N^ _O...N-NV.x */
    $"48E7 1F38 286E 000C 266E 0008 2054 3A28"      /* H..8(n..&n.. T:( */
    $"0016 594F 3F05 A9BF 245F 200A 6700 019C"      /* ..YO?.©ø$_ .g..ú */
    $"486E FFC4 A898 A89E 486E FFC0 A874 206E"      /* Hn.f®ò®ûHn.¿®t n */
    $"FFC0 3D68 0044 FFBE 206E FFC0 3628 004A"      /* .¿=h.D.æ n.¿6(.J */
    $"206E FFC0 3828 0048 206E FFC0 4868 0046"      /* n.¿8(.H n.¿Hh.F */
    $"486E FF7A 3F3C 0001 4EBA F8C2 1D6E FF99"      /* Hn.z?<..N∫.¬.n.ô */
    $"FFB9 4267 A887 3F3C 000C A88A 3F3C 0001"      /* .πBg®á?<..®ä?<.. */
    $"A889 422E FF98 7000 102E FF98 3F00 A888"      /* ®âB..òp....ô?.®à */
    $"486B FFE4 A88B 2F0A 3F3C FFFF A935 2054"      /* Hk..®â/.?<..©5 T */
    $"2768 001C FFEC 2F2B FFEC 4EBA F926 2054"      /* 'h..../+..N∫.& T */
    $"41E8 0008 43EE FFD6 22D8 22D8 206B FFEC"      /* A...C..+"ÿ"ÿ k.. */
    $"2050 41E8 000A 43EE FFDE 22D8 22D8 486E"      /* PA...C..."ÿ"ÿHn */
    $"FFDE 3F2E FFD8 3F2E FFD6 A8A8 3D6E FFE0"      /* ..?..ÿ?..+®®=n.. */
    $"FFE4 3D6E FFD8 FFE0 2054 7000 1028 0028"      /* ..=n.ÿ.. Tp..(.( */
    $"4A40 6F06 486E FFDE A8A4 3D6E FFD6 FFEC"      /* J@o.Hn..®§=n.+.. */
    $"206B FFEC 2050 302E FFD8 D068 000C 3D40"      /* k.. P0..ÿ-h..=@ */
    $"FFEE 486E FFEC A870 2F0A 2054 3F28 0012"      /* ..Hn..®p/. T?(.. */
    $"1F3C 0001 A945 594F 2F0A 3F2E FFEC 3F2E"      /* .<..©EYO/.?...?. */
    $"FFEE 2054 3F28 0012 A80B 2E1F 4206 2054"      /* .. T?(..®...B. T */
    $"BE68 0012 56C0 4A47 5EC1 C001 6716 2F0A"      /* æh..V¿JG^¡¿.g./. */
    $"2054 3F28 0012 4227 A945 2054 3147 0012"      /* T?(..B'©E T1G.. */
    $"7C01 600C 2F0A 2054 3F28 0012 4227 A945"      /* |.`./. T?(..B'©E */
    $"3F05 A936 2054 7000 1028 0028 4A40 6F06"      /* ?.©6 Tp..(.(J@o. */
    $"486E FFDE A8A4 4A06 670C 4267 2F0C 42A7"      /* Hn..®§J.g.Bg/.Bß */
    $"2F0B 4EBA F9CA 3F2E FFBE A887 3F03 A88A"      /* /.N∫. ?..æ®á?.®ä */
    $"3F04 A889 486E FFB9 486E FF7A 3F3C 0001"      /* ?.®âHn.πHn.z?<.. */
    $"4EBA F77A 1D6E FF99 FF78 7000 102E FF78"      /* N∫.z.n.ô.xp....x */
    $"3F00 A888 486E FFC4 A899 4CDF 1CF8 4E5E"      /* ?.®àHn.f®ôL...N^ */
    $"205F 504F 4ED0 4E56 FFE4 2F0C 286E 000E"      /* _PON-NV../.(n.. */
    $"42AE 0014 200C 6700 00AA 302E 000C 6000"      /* B®.. .g..™0...`. */
    $"008A 3F2E 0012 2F0C 2F2E 0008 2F0E 4EBA"      /* .ä?..././..../.N∫ */
    $"F95E 6000 008E 594F 3F2E 0012 2F0C 2F2E"      /* .^`..éYO?..././. */
    $"0008 2F0E 4EBA FD60 2D5F 0014 6000 0074"      /* ../.N∫.`-_..`..t */
    $"2F0C 4EBA F7AE 2F2E 0008 2054 4868 0008"      /* /.N∫.®/... THh.. */
    $"A8DF 2F0C 4EBA F7A6 6058 2F0C 4EBA F794"      /* ®./.N∫.¶`X/.N∫.1 */
    $"2054 70FF 2140 0020 2F0C 4EBA F790 6042"      /* Tp.!@. /.N∫.ê`B */
    $"2054 4AA8 001C 673A 2054 2D68 001C FFEC"      /* TJ®..g: T-h.... */
    $"2F2E FFEC 4EBA F762 2054 42A8 001C 6022"      /* /...N∫.b TB®..`" */
    $"2F0C 2F0E 4EBA FD96 6018 6700 FF76 5340"      /* /./.N∫.ñ`.g..vS@ */
    $"6784 5340 679A 5340 67B0 5340 67C2 5940"      /* gÑS@göS@g∞S@g¬Y@ */
    $"67DE 285F 4E5E 205F 4FEF 000C 4ED0"          /* g.(_N^ _O...N- */
};


resource 'CNTL' (133, "Popup menu") {
        {0, 0, 31, 31},
        1,
        invisible,
        100,
        1,
        2128,
        133,
```

```
        "Control for Dialog Simulation"
        "****** Extra bytes follow... ******/
        "* $"0000"                                              /* .. */
};

resource 'SIZE' (-1) {
        dontSaveScreen,
        acceptSuspendResumeEvents,
        enableOptionSwitch,
        canBackground,
        multiFinderAware,
        backgroundAndForeground,
        dontGetFrontClicks,
        ignoreChildDiedEvents,
        is32BitCompatible,
        reserved,
        reserved,
        reserved,
        reserved,
        reserved,
        reserved,
        reserved,
        1048576,
        1048576
};
```

10.5 NODDS DATA RETRIEVAL CODE

This section contains the FORTRAN source code and the MS-DOS batch files which CRC used on an IBM-PC compatible computer to use the Navy Oceanographic Data Distribution System (NODDS) to obtain atmospheric data for use with the BDPS program. Though this code is not a formal portion of the BDPS program, we have included it as an example of a means of providing wind data to BDPS from a source other than GRAM.

10.5.1 NODDS Batch Files

The two batch files (.BAT) included here install the support files for collecting and formatting the data and then configure the downloaded data files to support the production of wind table data for use with BDPS.

```
0001    copy wind.bat c:\nodds
0002    copy read.exe c:\nodds\fields
0003    copy getdir.exe c:\nodds\fields
0004    c:
0005    cd c:\nodds


0001    cd fields
0002    \nodds\makebat2
0003    getdir
0004    call d
0005    \nodds\decodfld
0006    rem
0007    copy c00*.??g alt1.tmp
0008    copy c20*.??g east1.tmp
0009    copy c21*.??g north1.tmp
```

```
0010     rem
0011     copy r00*.??g alt2.tmp
0012     copy r20*.??g east2.tmp
0013     copy r21*.??g north2.tmp
0014     rem
0015     copy d00*.??g alt3.tmp
0016     copy d20*.??g east3.tmp
0017     copy d21*.??g north3.tmp
0018     rem
0019     copy e00*.??g alt4.tmp
0020     copy e20*.??g east4.tmp
0021     copy e21*.??g north4.tmp
0022     rem
0023     copy f00*.??g alt5.tmp
0024     copy f20*.??g east5.tmp
0025     copy f21*.??g north5.tmp
0026     rem
0027     copy g00*.??g alt6.tmp
0028     copy g20*.??g east6.tmp
0029     copy g21*.??g north6.tmp
0030     rem
0031     copy h00*.??g alt7.tmp
0032     copy h20*.??g east7.tmp
0033     copy h21*.??g north7.tmp
0034     rem
0035     copy t00*.??g alt8.tmp
0036     copy t20*.??g east8.tmp
0037     copy t21*.??g north8.tmp
0038     rem
0039     copy i00*.??g alt9.tmp
0040     copy i20*.??g east9.tmp
0041     copy i21*.??g north9.tmp
0042     read
0043     copy wind1.dat b:east.dat
0044     copy wind2.dat b:north.dat
0045     del *.??g
0046     del *.tmp
0047     del d.bat
0048     del wind1.dat
0049     del wind2.dat
0050     del runbatch.bat
0051     cd ..
```

10.5.2 NODDS FORTRAN Files

These files were written to process the downloaded NODDS data into a
format suitable for transferring to the Macintosh for use with BDPS.

```
0001     C*******************************************************************C
0002     C                                                                 C
0003     C                                                                 C
0004     C     THIS PROGRAM EXTRACTS THE FIRST TWO LINES OF THE RUNBATCH.BAT   C
0005     C     FILE TO GET TO THE MAPTXT.DAT FILE.                          C
0006     C                                                                 C
0007     C                                                                 C
0008     C*******************************************************************C
0009     C
0010     C
0011           CHARACTER DIR*80
0012           OPEN(UNIT=1,STATUS='OLD',FILE='RUNBATCH.BAT')
0013           OPEN(UNIT=2,STATUS='NEW',FILE='D.BAT')
0014           DO I=1,2
```

```
0015          READ(1,10)DIR
0016          WRITE(2,10)DIR
0017          END DO
0018    10    FORMAT(A80)
0019          CLOSE(1)
0020          CLOSE(2)
0021          STOP
0022          END


0001    C*********************************************************************C
0002    C                                                                    C
0003    C                                                                    C
0004    C       THIS PROGRAM IS USED TO READ THE TEMPORARY FILES CREATED BY   C
0005    C                                                                    C
0006    C       WIND.BAT.  THE PROGRAM READS THESE FILES AND PUTS THE DATA IN C
0007    C                                                                    C
0008    C       A FORM THAT THE SUBROUTINE NOGAPS CAN READ.                   C
0009    C                                                                    C
0010    C*********************************************************************C
0011    C                                                                    C
0012    C                                                                    C
0013          DIMENSION A(1499),E(1499),N(1499),ALT(1499),EAST(1499)
0014          DIMENSION LONT(1499),LAT(1499),NORTH(1499)
0015          INTEGER I,M,L10,L20,L30,L40,MG,NG,J1
0016          REAL     ALT,LAT,LONT,EAST,NORTH,LAT0,LATB,LONT0,LONTR,SLAT,SLONT
0017          REAL     NUMBALT,MGRD,NGRD,MAT,A,E,N
0018          CHARACTER DIR1,DIR2,DIR3,DIR4
0019          CHARACTER*6 L11,L22,L33,L44
0020    C*********************************************************************C
0021    C                                                                    C
0022    C                                                                    C
0023    C       OPEN THE FILE THAT DEFINES THE AREA OF INTEREST
0024    C
0025    C
0026          OPEN(UNIT=78,STATUS='OLD',FILE='MAPTEMP.DAT')
0027    C
0028    C
0029    C       READ THE RANGE OF LATITUDES,LONGITUDES, AND WHAT QUADRANT(S)
0030    C       THEY EXIST.
0031    C
0032    C
0033          READ(78,6)
0034          READ(78,4000)L11,L22
0035          READ(78,4000)L33,L44
0036    4000  FORMAT(19X,A6,/,19X,A6)
0037          REWIND(78)
0038          READ(78,6)
0039          POS1=INDEX(L11,'.')
0040          POS2=INDEX(L22,'.')
0041          POS3=INDEX(L33,'.')
0042          POS4=INDEX(L44,'.')
0043          IF (POS1.LE.0)THEN
0044              READ(78,1)L10,DIR1
0045              LAT0=FLOAT(L10)
0046          ELSE IF (POS1.EQ.2)THEN
0047              READ(78,2)LAT0,DIR1
0048          ELSE IF (POS1.EQ.3)THEN
0049              READ(78,3)LAT0,DIR1
0050          END IF
0051          IF (POS2.LE.0)THEN
0052              READ(78,1)L20,DIR2
0053              LATB=FLOAT(L20)
0054          ELSE IF (POS2.EQ.2)THEN
```

337

```
0055              READ(78,2)LATB,DIR2
0056          ELSE IF (POS2.EQ.3)THEN
0057              READ(78,3)LATB,DIR2
0058          END IF
0059          IF (POS3.LE.0)THEN
0060              READ(78,1)L30,DIR3
0061              LONTR=FLOAT(L30)
0062          ELSE IF (POS3.EQ.2)THEN
0063              READ(78,2)LONTR,DIR3
0064          ELSE IF (POS3.EQ.3)THEN
0065              READ(78,3)LONTR,DIR3
0066          ELSE IF (POS3.EQ.4)THEN
0067              READ(78,4)LONTR,DIR3
0068          END IF
0069          IF (POS4.LE.0)THEN
0070              READ(78,1)L40,DIR4
0071              LONTO=FLOAT(L40)
0072          ELSE IF (POS4.EQ.2)THEN
0073              READ(78,2)LONTO,DIR4
0074          ELSE IF (POS4.EQ.3)THEN
0075              READ(78,3)LONTO,DIR4
0076          ELSE IF (POS4.EQ.4)THEN
0077              READ(78,4)LONTO,DIR4
0078          END IF
0079          DO I=1,2
0080          READ(78,6)
0081          END DO
0082          READ(78,7)MG,NG
0083          MGRD=FLOAT(MG)
0084          NGRD=FLOAT(NG)
0085      1   FORMAT(19X,I2,A1)
0086      2   FORMAT(19X,F3.1,A1)
0087      3   FORMAT(19X,F4.1,A1)
0088      4   FORMAT(19X,F5.1,A1)
0089      6   FORMAT(80X)
0090      7   FORMAT(9X,I2,8X,I2)
0091          MAT=MGRD*NGRD
0092      C
0093      C
0094      C   CALCULATE GRID SCALES
0095      C
0096      C
0097          IF (DIR1.EQ.'S')THEN
0098              LATO=-LATO
0099          END IF
0100          IF (DIR2.EQ.'S')THEN
0101              LATB=-LATB
0102          END IF
0103          IF (DIR3.EQ.'W')THEN
0104              LONTR=-LONTR
0105          END IF
0106          IF (DIR4.EQ.'W')THEN
0107              LONTO=-LONTO
0108          END IF
0109      C
0110          SLAT=-(LATO-LATB)/(NGRD-1)
0111          IF ((DIR3.EQ.'W').AND.(DIR4.EQ.'E'))THEN
0112              SLONT=((180-LONTO)+(180+LONTR))/(MGRD-1)
0113          ELSE
0114              SLONT=-(LONTO-LONTR)/(MGRD-1)
0115          END IF
0116      C
0117      C
0118      C***********************************************************************C
```

```
0119   C                                                              C
0120   C                                                              C
0121   C        READ THE 1000MB WIND DATA                             C
0122   C                                                              C
0123   C                                                              C
0124   C**********************************************************************C
0125   C
0126   C
0127         OPEN(UNIT=50,STATUS='OLD',FILE='ALT1.TMP')
0128         OPEN(UNIT=51,STATUS='OLD',FILE='EAST1.TMP')
0129         OPEN(UNIT=52,STATUS='OLD',FILE='NORTH1.TMP')
0130   C
0131         DO I=1,2
0132         READ(50,10)
0133         READ(51,10)
0134         READ(52,10)
0135         END DO
0136   C
0137         DO I=1,MAT,13
0138         READ(50,15)A(I),A(I+1),A(I+2),A(I+3),A(I+4),A(I+5),A(I+6),A(I+7),
0139        + A(I+8),A(I+9),A(I+10),A(I+11),A(I+12)
0140         READ(51,15)E(I),E(I+1),E(I+2),E(I+3),E(I+4),E(I+5),E(I+6),E(I+7),
0141        + E(I+8),E(I+9),E(I+10),E(I+11),E(I+12)
0142         READ(52,15)N(I),N(I+1),N(I+2),N(I+3),N(I+4),N(I+5),N(I+6),N(I+7),
0143        + N(I+8),N(I+9),N(I+10),N(I+11),N(I+12)
0144         END DO
0145   C
0146         CLOSE(50)
0147         CLOSE(51)
0148         CLOSE(52)
0149   C
0150   C
0151   C**********************************************************************C
0152   C                                                              C
0153   C                                                              C
0154   C        READ THE 925MB WIND DATA                              C
0155   C                                                              C
0156   C                                                              C
0157   C**********************************************************************C
0158   C
0159   C
0160         OPEN(UNIT=53,STATUS='OLD',FILE='ALT2.TMP')
0161         OPEN(UNIT=54,STATUS='OLD',FILE='EAST2.TMP')
0162         OPEN(UNIT=55,STATUS='OLD',FILE='NORTH2.TMP')
0163   C
0164         DO I=1,2
0165         READ(53,10)
0166         READ(54,10)
0167         READ(55,10)
0168         END DO
0169   C
0170         DO I=(MAT+1),(2*MAT),13
0171         READ(53,15)A(I),A(I+1),A(I+2),A(I+3),A(I+4),A(I+5),A(I+6),A(I+7),
0172        + A(I+8),A(I+9),A(I+10),A(I+11),A(I+12)
0173         READ(54,15)E(I),E(I+1),E(I+2),E(I+3),E(I+4),E(I+5),E(I+6),E(I+7),
0174        + E(I+8),E(I+9),E(I+10),E(I+11),E(I+12)
0175         READ(55,15)N(I),N(I+1),N(I+2),N(I+3),N(I+4),N(I+5),N(I+6),N(I+7),
0176        + N(I+8),N(I+9),N(I+10),N(I+11),N(I+12)
0177         END DO
0178   C
0179         CLOSE(53)
0180         CLOSE(54)
0181         CLOSE(55)
0182   C
```

```
0183    C
0184    C*************************************************************************C
0185    C                                                                        C
0186    C                                                                        C
0187    C        READ THE 850MB WIND DATA                                        C
0188    C                                                                        C
0189    C                                                                        C
0190    C*************************************************************************C
0191    C
0192    C
0193            OPEN(UNIT=56,STATUS='OLD',FILE='ALT3.TMP')
0194            OPEN(UNIT=57,STATUS='OLD',FILE='EAST3.TMP')
0195            OPEN(UNIT=58,STATUS='OLD',FILE='NORTH3.TMP')
0196    C
0197            DO I=1,2
0198            READ(56,10)
0199            READ(57,10)
0200            READ(58,10)
0201            END DO
0202    C
0203            DO I=(2*MAT+1),(3*MAT),13
0204            READ(56,15)A(I),A(I+1),A(I+2),A(I+3),A(I+4),A(I+5),A(I+6),A(I+7),
0205          + A(I+8),A(I+9),A(I+10),A(I+11),A(I+12)
0206            READ(57,15)E(I),E(I+1),E(I+2),E(I+3),E(I+4),E(I+5),E(I+6),E(I+7),
0207          + E(I+8),E(I+9),E(I+10),E(I+11),E(I+12)
0208            READ(58,15)N(I),N(I+1),N(I+2),N(I+3),N(I+4),N(I+5),N(I+6),N(I+7),
0209          + N(I+8),N(I+9),N(I+10),N(I+11),N(I+12)
0210            END DO
0211    C
0212            CLOSE(56)
0213            CLOSE(57)
0214            CLOSE(58)
0215    C
0216    C
0217    C*************************************************************************C
0218    C                                                                        C
0219    C                                                                        C
0220    C        READ THE 700MB WIND DATA                                        C
0221    C                                                                        C
0222    C                                                                        C
0223    C*************************************************************************C
0224    C
0225    C
0226            OPEN(UNIT=59,STATUS='OLD',FILE='ALT4.TMP')
0227            OPEN(UNIT=60,STATUS='OLD',FILE='EAST4.TMP')
0228            OPEN(UNIT=61,STATUS='OLD',FILE='NORTH4.TMP')
0229    C
0230            DO I=1,2
0231            READ(59,10)
0232            READ(60,10)
0233            READ(61,10)
0234            END DO
0235    C
0236            DO I=3*MAT+1,4*MAT,13
0237            READ(59,15)A(I),A(I+1),A(I+2),A(I+3),A(I+4),A(I+5),A(I+6),A(I+7),
0238          + A(I+8),A(I+9),A(I+10),A(I+11),A(I+12)
0239            READ(60,15)E(I),E(I+1),E(I+2),E(I+3),E(I+4),E(I+5),E(I+6),E(I+7),
0240          + E(I+8),E(I+9),E(I+10),E(I+11),E(I+12)
0241            READ(61,15)N(I),N(I+1),N(I+2),N(I+3),N(I+4),N(I+5),N(I+6),N(I+7),
0242          + N(I+8),N(I+9),N(I+10),N(I+11),N(I+12)
0243            END DO
0244    C
0245            CLOSE(59)
0246            CLOSE(60)
```

```
0247            CLOSE(61)
0248     C
0249     C
0250     C*******************************************************************C
0251     C                                                                  C
0252     C                                                                  C
0253     C        READ THE 500MB WIND DATA                                  C
0254     C                                                                  C
0255     C                                                                  C
0256     C*******************************************************************C
0257     C
0258     C
0259            OPEN(UNIT=62,STATUS='OLD',FILE='ALT5.TMP')
0260            OPEN(UNIT=63,STATUS='OLD',FILE='EAST5.TMP')
0261            OPEN(UNIT=64,STATUS='OLD',FILE='NORTH5.TMP')
0262     C
0263            DO I=1,2
0264            READ(62,10)
0265            READ(63,10)
0266            READ(64,10)
0267            END DO
0268     C
0269            DO I=4*MAT+1,5*MAT,13
0270            READ(62,15)A(I),A(I+1),A(I+2),A(I+3),A(I+4),A(I+5),A(I+6),A(I+7),
0271          + A(I+8),A(I+9),A(I+10),A(I+11),A(I+12)
0272            READ(63,15)E(I),E(I+1),E(I+2),E(I+3),E(I+4),E(I+5),E(I+6),E(I+7),
0273          + E(I+8),E(I+9),E(I+10),E(I+11),E(I+12)
0274            READ(64,15)N(I),N(I+1),N(I+2),N(I+3),N(I+4),N(I+5),N(I+6),N(I+7),
0275          + N(I+8),N(I+9),N(I+10),N(I+11),N(I+12)
0276            END DO
0277     C
0278            CLOSE(62)
0279            CLOSE(63)
0280            CLOSE(64)
0281     C
0282     C
0283     C*******************************************************************C
0284     C                                                                  C
0285     C                                                                  C
0286     C        READ THE 400MB WIND DATA                                  C
0287     C                                                                  C
0288     C                                                                  C
0289     C*******************************************************************C
0290     C
0291     C
0292            OPEN(UNIT=65,STATUS='OLD',FILE='ALT6.TMP')
0293            OPEN(UNIT=66,STATUS='OLD',FILE='EAST6.TMP')
0294            OPEN(UNIT=67,STATUS='OLD',FILE='NORTH6.TMP')
0295     C
0296            DO I=1,2
0297            READ(65,10)
0298            READ(66,10)
0299            READ(67,10)
0300            END DO
0301     C
0302            DO I=5*MAT+1,6*MAT,13
0303            READ(65,15)A(I),A(I+1),A(I+2),A(I+3),A(I+4),A(I+5),A(I+6),A(I+7),
0304          + A(I+8),A(I+9),A(I+10),A(I+11),A(I+12)
0305            READ(66,15)E(I),E(I+1),E(I+2),E(I+3),E(I+4),E(I+5),E(I+6),E(I+7),
0306          + E(I+8),E(I+9),E(I+10),E(I+11),E(I+12)
0307            READ(67,15)N(I),N(I+1),N(I+2),N(I+3),N(I+4),N(I+5),N(I+6),N(I+7),
0308          + N(I+8),N(I+9),N(I+10),N(I+11),N(I+12)
0309            END DO
0310     C
```

341

```
0311          CLOSE(65)
0312          CLOSE(66)
0313          CLOSE(67)
0314     C
0315     C
0316     C*******************************************************************C
0317     C                                                                  C
0318     C                                                                  C
0319     C          READ THE 300MB WIND DATA                                C
0320     C                                                                  C
0321     C                                                                  C
0322     C*******************************************************************C
0323     C
0324     C
0325          OPEN(UNIT=68,STATUS='OLD',FILE='ALT7.TMP')
0326          OPEN(UNIT=69,STATUS='OLD',FILE='EAST7.TMP')
0327          OPEN(UNIT=70,STATUS='OLD',FILE='NORTH7.TMP')
0328     C
0329          DO I=1,2
0330          READ(68,10)
0331          READ(69,10)
0332          READ(70,10)
0333          END DO
0334     C
0335          DO I=6*MAT+1,7*MAT,13
0336          READ(68,15)A(I),A(I+1),A(I+2),A(I+3),A(I+4),A(I+5),A(I+6),A(I+7),
0337         + A(I+8),A(I+9),A(I+10),A(I+11),A(I+12)
0338          READ(69,15)E(I),E(I+1),E(I+2),E(I+3),E(I+4),E(I+5),E(I+6),E(I+7),
0339         + E(I+8),E(I+9),E(I+10),E(I+11),E(I+12)
0340          READ(70,15)N(I),N(I+1),N(I+2),N(I+3),N(I+4),N(I+5),N(I+6),N(I+7),
0341         + N(I+8),N(I+9),N(I+10),N(I+11),N(I+12)
0342          END DO
0343     C
0344          CLOSE(68)
0345          CLOSE(69)
0346          CLOSE(70)
0347     C
0348     C
0349     C*******************************************************************C
0350     C                                                                  C
0351     C                                                                  C
0352     C          READ THE 250MB WIND DATA                                C
0353     C                                                                  C
0354     C                                                                  C
0355     C*******************************************************************C
0356     C
0357     C
0358          OPEN(UNIT=71,STATUS='OLD',FILE='ALT8.TMP')
0359          OPEN(UNIT=72,STATUS='OLD',FILE='EAST8.TMP')
0360          OPEN(UNIT=73,STATUS='OLD',FILE='NORTH8.TMP')
0361     C
0362          DO I=1,2
0363          READ(71,10)
0364          READ(72,10)
0365          READ(73,10)
0366          END DO
0367     C
0368          DO I=7*MAT+1,8*MAT,13
0369          READ(71,15)A(I),A(I+1),A(I+2),A(I+3),A(I+4),A(I+5),
0370         + A(I+6),A(I+7),A(I+8),A(I+9),A(I+10),A(I+11),A(I+12)
0371          READ(72,15)E(I),E(I+1),E(I+2),E(I+3),E(I+4),E(I+5),E(I+6),E(I+7),
0372         + E(I+8),E(I+9),E(I+10),E(I+11),E(I+12)
0373          READ(73,15)N(I),N(I+1),N(I+2),N(I+3),N(I+4),N(I+5),N(I+6),N(I+7),
0374         + N(I+8),N(I+9),N(I+10),N(I+11),N(I+12)
```

```
0375           END DO
0376      C
0377           CLOSE(71)
0378           CLOSE(72)
0379           CLOSE(73)
0380      C
0381      C
0382      C*****************************************************************C
0383      C                                                                 C
0384      C                                                                 C
0385      C      READ THE 200MB WIND DATA                                   C
0386      C                                                                 C
0387      C                                                                 C
0388      C*****************************************************************C
0389      C
0390      C
0391           OPEN(UNIT=74,STATUS='OLD',FILE='ALT9.TMP')
0392           OPEN(UNIT=75,STATUS='OLD',FILE='EAST9.TMP')
0393           OPEN(UNIT=76,STATUS='OLD',FILE='NORTH9.TMP')
0394      C
0395           DO I=1,2
0396           READ(74,10)
0397           READ(75,10)
0398           READ(76,10)
0399           END DO
0400      C
0401           DO I=8*MAT+1,9*MAT,13
0402           READ(74,15)A(I),A(I+1),A(I+2),A(I+3),A(I+4),A(I+5),A(I+6),A(I+7),
0403          + A(I+8),A(I+9),A(I+10),A(I+11),A(I+12)
0404           READ(75,15)E(I),E(I+1),E(I+2),E(I+3),E(I+4),E(I+5),E(I+6),E(I+7),
0405          + E(I+8),E(I+9),E(I+10),E(I+11),E(I+12)
0406           READ(76,15)N(I),N(I+1),N(I+2),N(I+3),N(I+4),N(I+5),N(I+6),N(I+7),
0407          + N(I+8),N(I+9),N(I+10),N(I+11),N(I+12)
0408           END DO
0409      C
0410           CLOSE(74)
0411           CLOSE(75)
0412           CLOSE(76)
0413      C
0414      10   FORMAT(80X)
0415      15   FORMAT(13F6.0)
0416      C
0417      C*****************************************************************C
0418      C                                                                 C
0419      C                                                                 C
0420      C      CONVERT INTEGER DATA TO REAL AND SCALE                     C
0421      C                                                                 C
0422      C                                                                 C
0423      C*****************************************************************C
0424      C
0425      C
0426           SCALE=1
0427           DO J=1,9
0428               LONT((J-1)*MAT+1)=LONT0
0429               LAT((J-1)*MAT+1)=LAT0
0430           DO 20 I=(J-1)*MAT+1,J*MAT
0431               ALT(I)=A(I)/30.0
0432               EAST(I)=E(I)/19.44
0433               NORTH(I)=N(I)/19.44
0434      C
0435               IF (I.EQ.SCALE*MGRD) THEN
0436                   LONT(I+1)=LONT0
0437                   LAT(I+1)=LAT(I)+SLAT
0438                   SCALE=SCALE+1
```

343

```
0439                    ELSE
0440                        LONT(I+1)=LONT(I)+SLONT
0441                        LAT(I+1)=LAT(I)
0442                    END IF
0443                    IF (LAT(I+1).GT.90.0)THEN
0444                        LAT(I+1)=LAT(I+1)-180.0
0445                    ELSE IF (LAT(I-1).LT.-90.0)THEN
0446                        LAT(I+1)=LAT(I+1)+180.0
0447                    END IF
0448                    IF (LONT(I+1).GT.180.0)THEN
0449                        LONT(I+1)=LONT(I+1)-360.0
0450                    ELSE IF (LONT(I+1).LT.-180.0)THEN
0451                        LONT(I+1)=LONT(I+1)+360.0
0452                    END IF
0453     20         CONTINUE
0454            END DO
0455     C
0456     C*****************************************************************************C
0457     C                                                                            C
0458     C                                                                            C
0459     C         SORT WIND DATA ON LATITUDE                                         C
0460     C                                                                            C
0461     C                                                                            C
0462     C*****************************************************************************C
0463     C
0464     .
0465            DO 200 I=1,9*MAT-1
0466                M=I+1
0467                DO 100 J=M,9*MAT
0468                  IF (LAT(I) .GT. LAT(J)) THEN
0469                      TEMP1=ALT(I)
0470                      TEMP3=LAT(I)
0471                      TEMP4=LONT(I)
0472                      TEMP5=EAST(I)
0473                      TEMP6=NORTH(I)
0474                      ALT(I)=ALT(J)
0475                      LAT(I)=LAT(J)
0476                      LONT(I)=LONT(J)
0477                      EAST(I)=EAST(J)
0478                      NORTH(I)=NORTH(J)
0479                      ALT(J)=TEMP1
0480                      LAT(J)=TEMP3
0481                      LONT(J)=TEMP4
0482                      EAST(J)=TEMP5
0483                      NORTH(J)=TEMP6
0484                  END IF
0485     100      CONTINUE
0486     200      CONTINUE
0487     C
0488     C
0489     C*****************************************************************************C
0490     .                                                                            C
0491     C                                                                            C
0492     C         SORT DATA ON LONGITUDE                                             C
0493     C                                                                            C
0494     C                                                                            C
0495     C*****************************************************************************C
0496     C
0497     C
0498            DO 310 J1=1,NGRD
0499            DO 210 I=(J1-1)*MGRD*9+1,J1*MGRD*9-1
0500                M=I+1
0501                DO 110 J=M,J1*MGRD*9
0502                  IF (LONT(I) .GT. LONT(J)) THEN
```

```
             TEMP1=ALT(I)
             TEMP3=LAT(I)
             TEMP4=LONT(I)
             TEMP5=EAST(I)
             TEMP6=NORTH(I)
             ALT(I)=ALT(J)
             LAT(I)=LAT(J)
             LONT(I)=LONT(J)
             EAST(I)=EAST(J)
             NORTH(I)=NORTH(J)
             ALT(J)=TEMP1
             LAT(J)=TEMP3
             LONT(J)=TEMP4
             EAST(J)=TEMP5
             NORTH(J)=TEMP6
          END IF
0519  110    CONTINUE
0520  210    CONTINUE
0521  310    CONTINUE
0522         DO 610 J1=1,9*MAT,9
0523         DO 510 I=(J -1)+1,(J1-1)+8
0524          M=I+1
0525           DO 410 J=M,(J1-1)+9
0526            IF (ALT(I) .GT. ALT(J)) THEN
0527                TEMP1=ALT(I)
0528                TEMP3=LAT(I)
0529                TEMP4=LONT(I)
0530                TEMP5=EAST(I)
0531                TEMP6=NORTH(I)
0532                ALT(I)=ALT(J)
0533                LAT(I)=LAT(J)
0534                LONT(I)=LONT(J)
0535                EAST(I)=EAST(J)
0536                NORTH(I)=NORTH(J)
0537                ALT(J)=TEMP1
0538                LAT(J)=TEMP3
0539                LONT(J)=TEMP4
0540                EAST(J)=TEMP5
0541                NORTH(J)=TEMP6
0542            END IF
0543  410        CONTINUE
0544  510    CONTINUE
0545  610    CONTINUE
0546  C
0547  C
0548  C*********************************************************************C
0549  C                                                                    C
0550  C                                                                    C
0551  C      WRITE THE DATA TO WIND1.DAT ( EAST WIND VECTOR )              C
0552  C      WRITE THE DATA TO WIND2.DAT ( NORTH WIND VECTOR )             C
0553  C                                                                    C
0554  C                                                                    C
0555  C*********************************************************************C
0556  C
0557  C
0558         OPEN(UNIT=77,STATUS='NEW',FILE='WIND1.DAT')
0559         OPEN(UNIT=79,STATUS='NEW',FILE='WIND2.DAT')
0560  C
0561  C
0562         NUMBALT=9.0
0563         WRITE(77,51)NUMBALT,MGRD,NGRD
0564         WRITE(77,*)
0565         WRITE(77,52)(ALT(I),I=1,9)
0566         WRITE(77,*)
```

345

```
0567              WRITE(77,52) (LONT(I),I=1,9*MGRD,9)
0568              WRITE(77,*)
0569              WRITE(77,52) (LAT(I),I=1,9*MAT,9*MGRD)
0570              WRITE(77,*)
0571      C
0572              WRITE(79,51) NUMBALT,MGRD,NGRD
0573              WRITE(79,*)
0574              WRITE(79,52) (ALT(I),I=1,9)
0575              WRITE(79,*)
0576              WRITE(79,52) (LONT(I),I=1,9*MGRD,9)
0577              WRITE(79,*)
0578              WRITE(79,52) (LAT(I),I=1,9*MAT,9*MGRD)
0579              WRITE(79,*)
0580      C
0581                  DO J=1,9
0582                  WRITE(77,50) (EAST(I),I=(J-1)*MAT+1,MAT*J)
0583                  WRITE(79,50) (NORTH(I),I=(J-1)*MAT+1,MAT*J)
0584                  END DO
0585      C
0586      50    FORMAT(F8.3,F8.3,F8.3,F8.3)
0587      51    FORMAT(F8.3,F8.3,F8.3)
0588      52    FORMAT(F9.3,F9.3,F9.3,F9.3)
0589      C
0590      C*******************************************************************************C
0591      C                                                                              C
0592      C                                                                              C
0593      C        CLOSE OUTPUT FILES                                                    C
0594      C                                                                              C
0595      C                                                                              C
0596      C*******************************************************************************C
0597      C
0598      C
0599              CLOSE(77)
0600              CLOSE(79)
0601              STOP
0602              END
```

20.0 **BDPS USER MANUAL**

The purpose of this appendix to the final report is to describe the procedures with which a user may operate the Balloon Drift Pattern Simulation (BDPS).

20.1 BDPS CONCEPT OF OPERATIONS

CRC designed the BDPS tool to be easily operated for the analysis of a balloon's drift pattern as influenced primarily by atmospheric winds. BDPS was written for and tested on Macintosh computers with either 68020 or 68030 processors (and 68881 or 68882 floating point coprocessors). In general, a user supplies certain files according to the desired BDPS modeling options. The user starts the BDPS application and then takes a series of steps to generate, display, and save graphical representations of balloon drift patterns. The following sections describe the files involved and the steps to be taken to produce drift patterns with BDPS.

20.2 REQUIRED FILES FOR BDPS OPERATION

Besides the BDPS application program, several files are required for the successful operation of BDPS. Every BDPS run will require an ascent profile which contains data that represents a particular balloon configuration's vertical ascent rate as a function of time. Then, depending on the user's preference, the BDPS application will either require a set of climate files associated with the GRAM atmosphere model or a set of files for using the BDPS wind table option. These files are discussed in the following sections.

20.2.1 **Ascent Profile**

Every BDPS run will require an ascent profile which contains data that represents a particular balloon configuration's vertical ascent rate as a function of time. The file should be named "Ascent Profile" and should be located in the same Macintosh folder as the BDPS application. The file may be created with any text editor or word processor which is capable of saving the ascent profile in a "text-only" format.

The ascent profile of a balloon configuration is constructed as a number of data points which represent discrete samples from the complete curve which describes the entire vertical motion profile. Because the BDPS program will perform linear interpolation between successive points, the points should be chosen to make a linear interpolation valid. The ascent profile will consist of two columns of numbers where the first column is the time reference point and the

second column is the vertical velocity corresponding to that particular time. Figure 20.2.1-1 shows the format to be followed in the creation of an ascent profile.

First Text Line: Time 1 (s) Vertical Velocity at Time 1 (m/s)
Second Text Line: Time 2 (s) Vertical Velocity at Time 2 (m/s)
(Additional Lines)

Figure 20.2.1-1. Ascent Profile Format

20.2.2 Climate-Related Files

If the BDPS user desires to use the "climate" model option for providing wind data, then a number of files must be present within the same Macintosh folder as the BDPS application. These files are distributed on magnetic media with the BDPS application and should not be altered in content, nor should the file names be changed. Table 20.2.2-1 lists the files and gives a brief description of each file.

Table 20.2.2-1. Climate-Related Files

NAME OF DATA FILE	DESCRIPTION OF FILE CONTENTS
NASPGROVES.F	Groves data
NASPPPWCS.F	Density-velocity correlations & large scale fraction data
NASPQBO.F	Quasi-biennial oscillation data
NASPRRW.F	Random perturbation data
NASPSP.F	Spherical harmonic data
NMC.DAT	National Meteorological Center grid data
M1.DAT	Meteorological data for month of January
M2.DAT	Meteorological data for month of February
M3.DAT	Meteorological data for month of March
M4.DAT	Meteorological data for month of April
M5.DAT	Meteorological data for month of May
M6.DAT	Meteorological data for month of June
M7.DAT	Meteorological data for month of July
M8.DAT	Meteorological data for month of August
M9.DAT	Meteorological data for month of September
M10.DAT	Meteorological data for month of October
M11.DAT	Meteorological data for month of November
M12.DAT	Meteorological data for month of December

20.2.3 Wind Table Files

If the BDPS user wishes to produce a balloon drift pattern as a function of a wind model other than the GRAM climate model, the user should provide two

data files: "EAST.DAT" and "NORTH.DAT." These files describe, respectively, the east and north wind velocity components as functions of latitude, longitude, and altitude. The files should be located in the same Macintosh folder as the BDPS application. Figure 20.2.3-1 shows the format for each of the two files.

```
<no. of altitude values> <no. of longitude values> <no of latitude
     values>

<blank line>

Altitude_1    Altitude_2    ...    Altitude_k

<blank line>

Longitude_1   Longitude_2   ...    Longitude_m

<blank line>

Latitude_1    Latitude_2    ...    Latitude_n

<blank line>

<wind magnitude values:  three-dimensional nested loop with outermost
     index for latitude dependency, middle index for longitude values,
     then the innermost index for altitude values.>
```

Figure 20.2.3.   Wind Table File Format

20.3   BDPS OPERATION PROCEDURE

This section describes the steps to take in the operation of BDPS to produce drift patterns. The figures in this section are screen snapshots from actual operation of the BDPS program. The procedure discussion assumes that the BDPS user already has familiarity with Macintosh procedures.

After double-clicking the BDPS application, the user must choose whether to open an existing "mission" file or to start with a new mission file. A saved mission file contains the user's run setup parameters and the data for making plots onscreen. For the sake of discussion, the user has selected "New Mission" from the File menu and will now see the run setup dialog as shown in Figure 20.3-1.

```
┌─                    Edit                                    ⌐?⌐     ─┐
│╔═══════════════════════════════════════════════════════════════════╗│
│║                                                                   ║│
│║  ¹ission  ┌─────────────────────────────────────────────────────┐ ║│
│║  Label:   │Wallops Island flight - 2 configuration (this text may be│ ║│
│║           │used to identify the mission represented by this data) │ ║│
│║           └─────────────────────────────────────────────────────┘ ║│
│║  Launch      Latitude:   ┌──────┐   deg                          ║│
│║  Position                │37.9  │                                ║│
│║              Longitude:  ┌──────┐   ⦿ deg West   ○ deg East      ║│
│║                          │75.5  │                                ║│
│║         Initial Altitude: ┌──────┐  ⦿ m   ○ km                   ║│
│║                          │1.0   │                                ║│
│║         Flight Duration: ┌──────┐  ○ sec ○ min ⦿ hr             ║│
│║                          │24.0  │                                ║│
│║                                                                   ║│
│║              Wind Model: ┌──────────────────┐      ┌───────────┐ ║│
│║                          │ January Climate  ▼│      │  Run  ↖  │ ║│
│║      Input Ascent Profile:┌──────────────────┐     └───────────┘ ║│
│║                          │ Ascent Profile   ▼│     ┌───────────┐ ║│
│║                                                     │  Save     │ ║│
│║                                                     └───────────┘ ║│
│║                                                     ┌───────────┐ ║│
│║                                                     │  Map      │ ║│
│║                                                     └───────────┘ ║│
│║                                                     ┌───────────┐ ║│
│║                                                     │  Close     │ ║│
│║                                                     └───────────┘ ║│
│╚═══════════════════════════════════════════════════════════════════╝│
└───────────────────────────────────────────────────────────────────┘
```

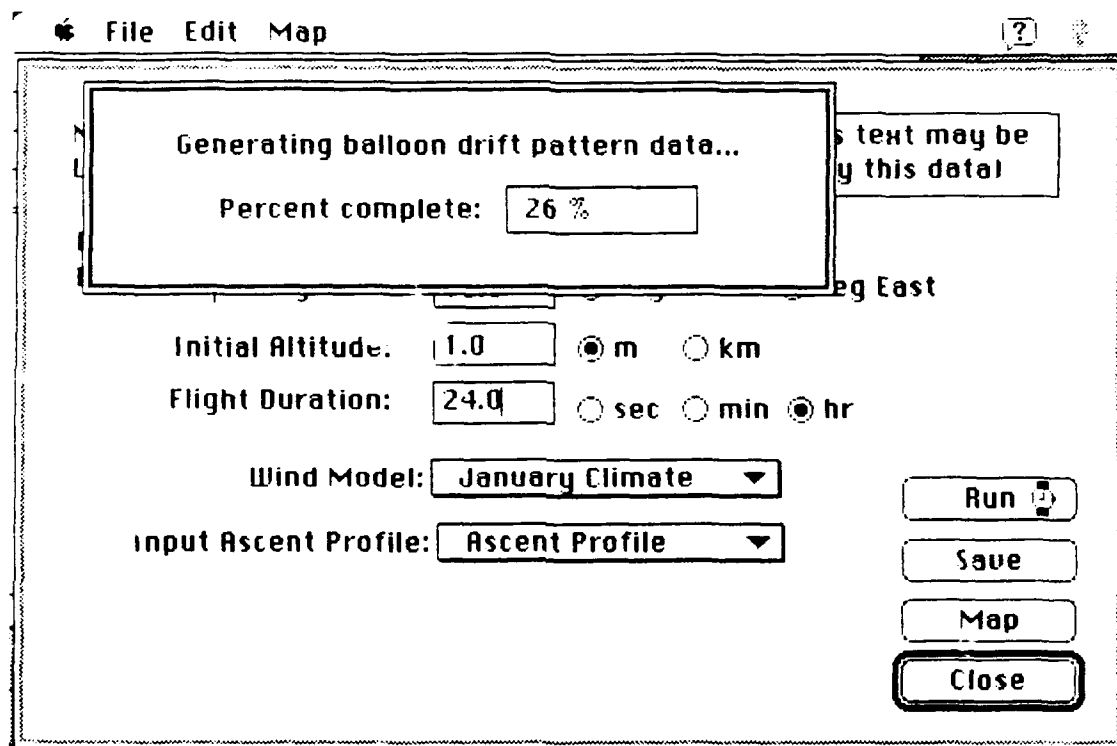**Figure 20.3-1.   Sample BDPS Run Setup Dialog**

At this point, the user may enter a set of numbers in the various boxes and may select the desired units for the numbers. Table 20.3-1 lists the valid values for the various number entry boxes. If an invalid entry is made, BDPS will beep and highlight the invalid entry when the user tries to run or save with the invalid entry.

Table 20.3-1.   Run Setup Ranges of Values

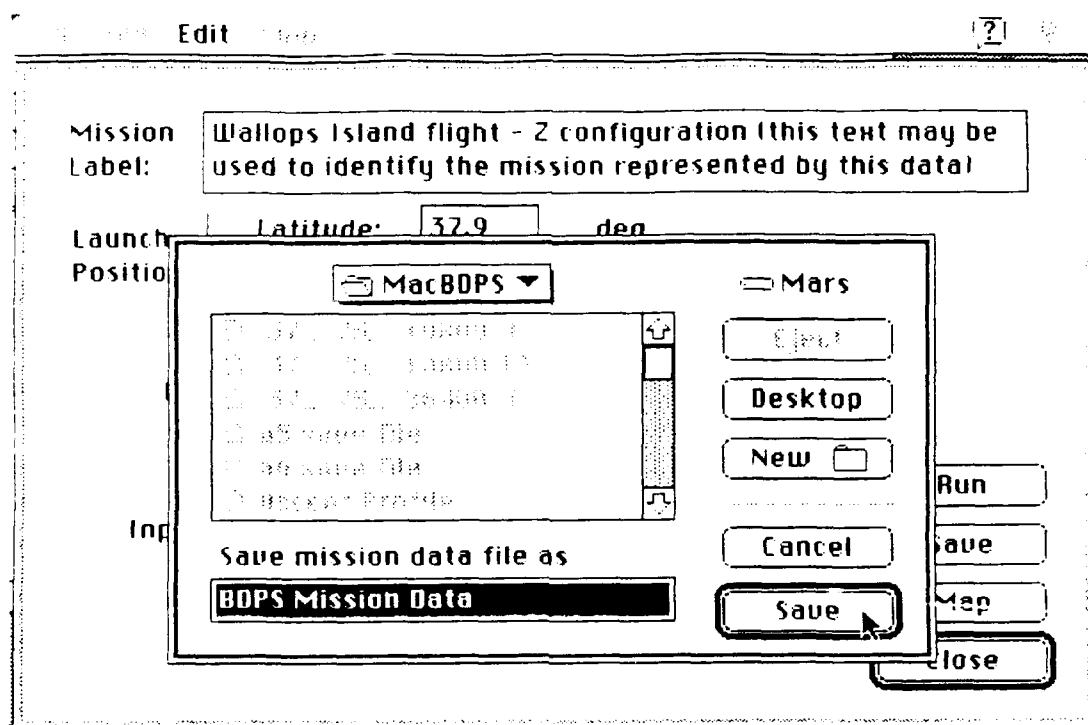| Entry Field	Range of Acceptable Values
Latitude	-90.0 to 90.0 degrees
Longitude	-180.0 to 180.0 degrees
Altitude	0.0 to 1,000,000.0 meters
Duration	0.0 to 720.0 hours

After entering the desired values, the user should use the four buttons in the lower-right dialog corner to take action within BDPS. The recommended sequence is top-to-bottom: run, save, map, close. When the user clicks the run button, BDPS will then generate the drift pattern data for the specified mission. During this stage, BDPS reports its progress as shown in Figure 20.3-2.

Occasionally, the status will not be updated for several moments while the climate model engages in intensive file access activity to establish or re-establish grids of data from which to interpolate.
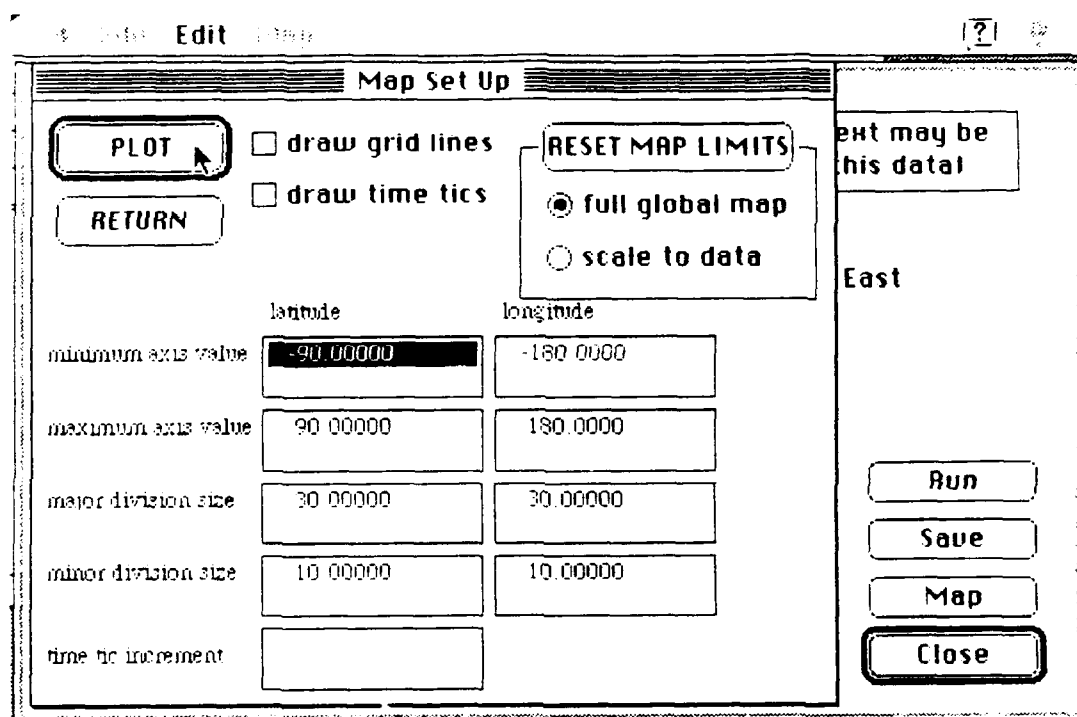

Figure 20.3-2.   Drift Pattern Data Generation Progress Dialog

Once the run is complete, the user should save the data which was produced. This is initiated by clicking the Save button. The user will then have the chance to specify a file name for the mission file to be saved. This operation uses the standard Macintosh interface for file operations as shown in Figure 20.3-3.

Figure 20.3-3.   Saving the BDPS Mission File

Now the BDPS user may display the generated drift pattern onscreen by clicking the Map button.   BDPS will present a dialog box for setting up the onscreen plot. This configuration dialog is shown in Figure 20.3-4.
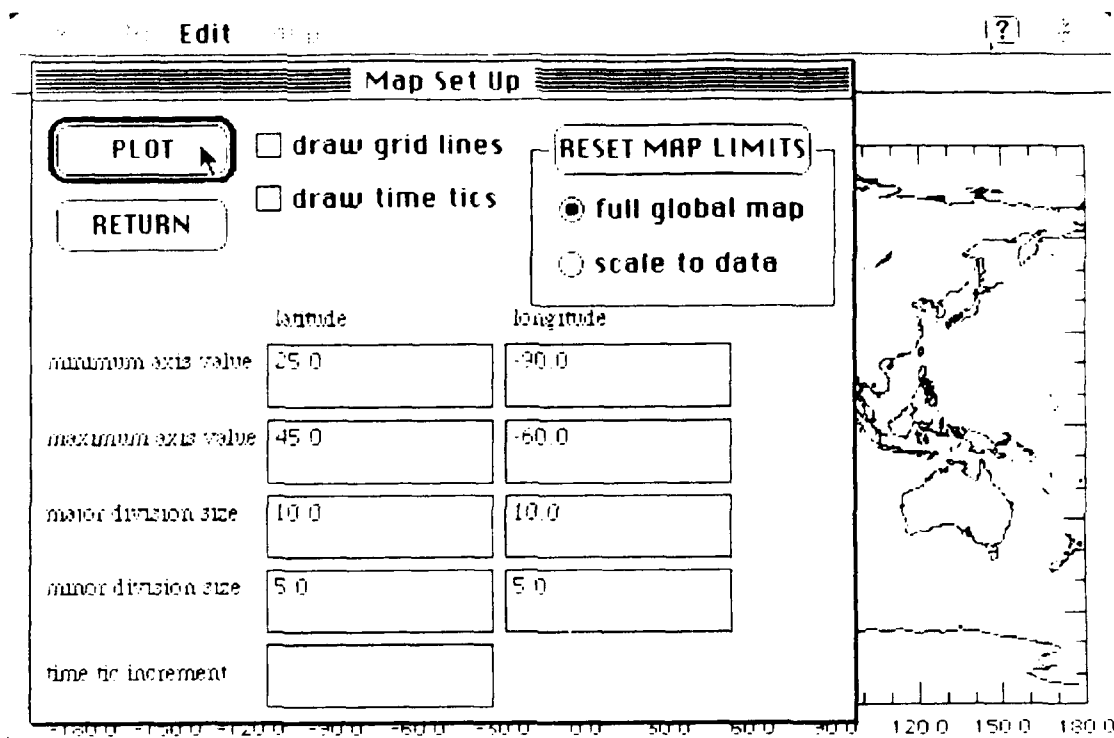
Figure 20.3-4. Setting Up the Drift Pattern Display

Once the user selects the Plot button, the screen will be cleared and the drift pattern will be displayed. BDPS produces outlines of major land masses first, and then uses a double-wide line to depict the path followed by the balloon in its trajectory. At this point, the Map menu is now active and may be used to save the map, to refresh the map, or to open a saved mission file to produce plots with previously generated data. Figure 20.3-5 shows a global map with a balloon drift originating from Wallops Island, Virginia. In that figure, the cursor is pointing to the balloon's trajectory path.
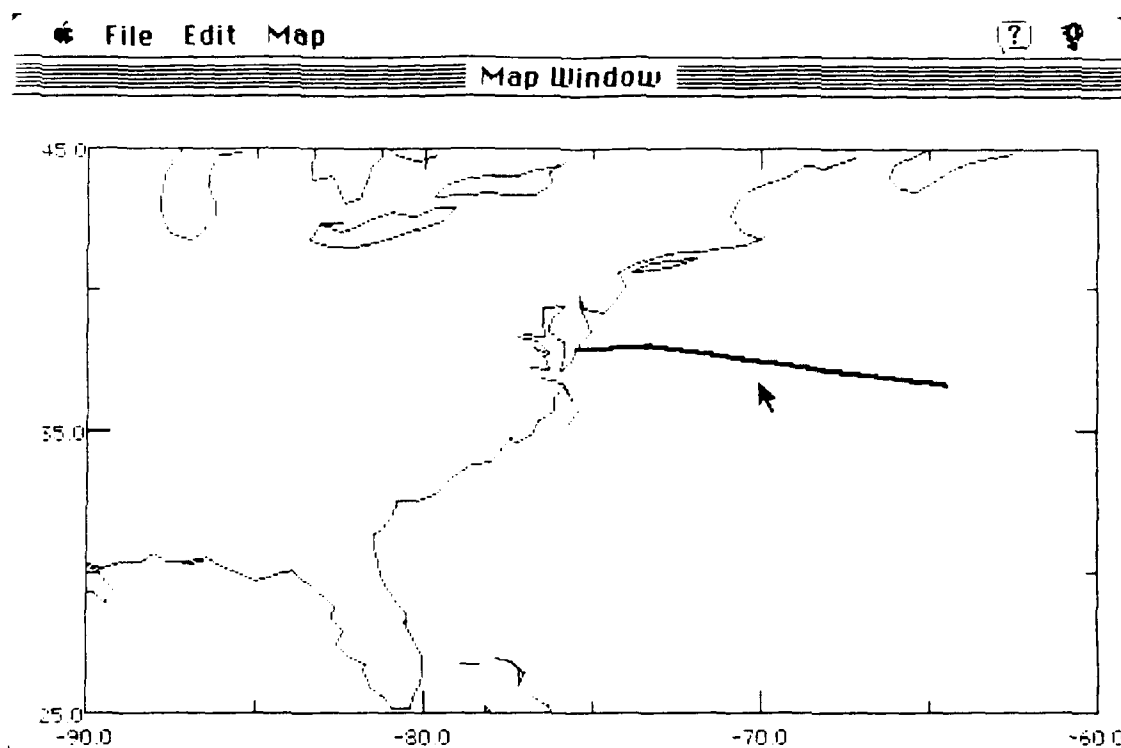
Figure 20.3-5.   Sample Drift Pattern

The drift patten in the Figure 20.3-5 is a good example of a case in which the user may wish to narrow the scope of the display and "zoom in" on the region of interest.   By selecting New Map from the Map menu, the user may change the map setup parameters as shown in Figure 20.3-6 and then produce a display as shown in Figure 20.3-7, which provides a clearer picture of the balloon's drift trajectory.

Figure 20.3-6. Providing New Map Parameters

Figure 20.3-7. Sample Regional Drift Pattern

Any drift pattern plot, such as in Figure 20.3-5 or Figure 20.3-7, may be saved as a Macintosh PICT file by using the Save option from the Map menu. The user may then choose the Done option from the Map menu and the Close button from the run setup dialog. At this point, the user may Quit the application or begin with New Mission or Open Mission. After quitting BDPS, the user may start a graphics program (e.g., MacDraw II) and then open the saved PICT file which contains an image of the drift pattern display. Depending on its capabilities, the graphics program may be used to annotate and/or print the drift pattern.